Theoretical Computer Science 77 (1990) 161-194
North-Holland



# DATA TYPES OVER MULTIPLE-VALUED LOGICS*

Don PIGOZZI

*Department of Mathematics, Iowa State University, Ames, IA 50011, USA*

**Abstract.** Let $\mathscr{L}$ be a multiple-valued logic. An *equality-test algebra over* $\mathscr{L}$ is a many-sorted algebra that contains the truth values of $\mathscr{L}$ in a special logic domain; it also includes certain operations that take values in the logic domain and are to be viewed as the characteristic functions of nonclassical predicates over the universe of the algebra. In particular the algebra contains for each sort $s$ the characteristic function of the nonclassical equality predicate over the domain of sort $s$. Nonclassical data structures of this form arise naturally in various computer science contexts. The basic properties of nonclassical equality-test algebras are investigated, and an axiomatization of their conditional-equational theory is obtained for a large class of multiple-valued logics. Some consequences for the specification of nonclassical equality-test data types are drawn.

## Introduction

Since the pioneering work of J. V. Guttag, S. N. Zilles, and the ADJ group algebraic methods have played an important role in software development. At the risk of greatly oversimplifying matters, one can describe the algebraic theory of data types as the theory of specification of data types by means of equations; in short, the equational logic of data structures. (By an *equation* in this Introduction we also mean a conditional equation.) The equational (and conditional equational) specification of data types has many important advantages over other specification methods and has been extensively developed. The best known advantage is the existence of initial objects. In fact this property turns out to characterize conditional-equational model classes [25, 26]. (Although final algebras do not always exist, they seem to be more easily handled in equational logic.) Initial algebras provide an effective means of selecting a canonical model of a set of equations. Another advantage is that the proof theory of equational logic is substantially simpler than that of more expressive languages, and has been effectively used as a means of implementing data types and even general purpose programming languages; see for instance [13]. Equational specification also has some disadvantages. The main one being the inherent limitation on the expressive power of equational languages. There are many data structures whose fundamental properties cannot be described, at least in a natural way, by means of equations. One way around this problem is to take the approach of classical algebraic logic and attempt to interpret the richer language within equational logic. In the ideal situation the interpretation can be carried out in the following way: the underlying logic of the richer language is embodied in a specific algebraic structure, which is then physically incorporated in

* Partially supported by NSF Grant DMS-8805870.

the data structure. The sentences of the more expressive language are then uniformly translated into equations (or finite systems of equations) in such a way that the data structure models a sentence iff it models the corresponding equation. This process is well known and widely applied in the case of the classical first-order predicate logic and homogeneous structures. The universal fragment of predicate logic over heterogeneous data structures is studied in some detail in [33]. In the present paper we will study the process in a more general context and apply it to a wide class of nonclassical logics.

The importance of nonclassical logics in computer science, and in particular in programming language semantics, has been recognized for some time. Three-valued logic arises naturally from the fact that a computation process may not terminate on some inputs, or that an operation on a data structure may not have a natural definition for certain data objects. On the other hand, four-valued logic seems natural when an attempt is made to formalize the theory of a database in which both the under-determination and the over-determination of data is allowed. Even infinite-valued logics prove useful in formalizing a process whereby a data object, such as a function with an infinite domain, is successively approximated by partially computed subobjects. Several specific examples of multiple-valued logics that have occurred in the literature in this context will be considered.

In Section 1 the process of algebraizing the universal fragment of the classical first-order theory of data types is reviewed; the main purpose here is to motivate subsequent work. Nonclassical logics in a very broad setting are discussed in Section 2 and the algebraization process is generalized in Section 3. It produces results that closely parallel those of the classical case for certain special logics. A large class of them, the *autoalgebraizable logics*, are investigated in Section 4. The classical algebraization process extends smoothly to these logics provided the deduction theorem holds. The results are presented in Section 5. In the main result of the paper, Theorem 5.3, we obtain an axiomatization of the conditional-equational theory of data structures that have been enriched by equality-test operations that take values in an autoalgebraizable logic. Some interesting consequences for the specification of data types are presented in Theorems 5.10 and 5.11.

The theory of nonclassical data types presented in this paper has to be considered incomplete because it fails to include within its scope most of the important nonclassical logics that arise in computer science. These logics are not autoalgebraizable, and the algebraization of the universal first-order theory of data structures over them is problematical. However we hope this paper represents a significant first step by clarifying the limitations of the methods of classical algebraic logic. The situation is discussed in more detail at the end of the paper in Section 6.

## Notation and terminology

We explain here some of the special notation and terminology used in the paper. For all unexplained notation we refer the reader to [14] or [33].

We will use the term *signature* in the familiar sense of [14]; it will mean a set of operation symbols, along with their associated types, over some set of sorts. By a *relational signature* we will mean a signature $\Sigma$ which contains predicate as well as operation symbols. (An ordinary signature is occasionally referred to as *algebraic* to emphasis the fact it is not relational.) We denote the sets of operation and relation symbols of $\Sigma$ respectively by $\Sigma_O$ and $\Sigma_P$. Let $S$ be the set of sorts of $\Sigma$. Then a *relational structure* $\boldsymbol{R}$ *of signature* $\Sigma$ (more simply a $\Sigma$-*structure*) is made up of an $S$-sorted *universe* (or *carrier*) $R = \langle R_s : s \in S \rangle$ together with an interpretation $\sigma^{\boldsymbol{R}}$ of each $\sigma \in \Sigma$. If $\sigma \in \Sigma_O$ and is of type $s_0 \dots s_{n-1} \to s$, then $\sigma^{\boldsymbol{R}} : R_{s_0} \times R_{s_1} \times \cdots \times R_{s_{n-1}} \to R_s$. If $\sigma \in \Sigma_P$ and is of type $s_0 \dots s_{n-1}$, then $\sigma^{\boldsymbol{R}} \subseteq R_{s_0} \times R_{s_1} \times \cdots \times R_{s_{n-1}}$. $R_s$ is called the *domain of sort*, or *type*, $s$ (or simply the *s-domain*) of $\boldsymbol{R}$.

By a *subsignature* of $\Sigma$ we mean a signature $\Sigma'$ with sort-set $S' \subseteq S$ such that

$$\Sigma'_{O_{s_0 \dots s_{n-1} \to s}} \subseteq \Sigma_{O_{s_0 \dots s_{n-1} \to s}}, \tag{1}$$

$$\Sigma'_{P_{s_0 \dots s_{n-1}}} \subseteq \Sigma_{P_{s_0 \dots s_{n-1}}}, \tag{2}$$

for all $s_0, \dots, s_{n-1}, s \in S'$. If $\boldsymbol{R}$ is a $\Sigma$-structure, then $\boldsymbol{R}_{\Sigma'}$ denotes the $\Sigma'$-structure with universe $R' = \langle R_s : s \in S' \rangle$ and operations and predicates $\sigma^{\boldsymbol{R}_{\Sigma'}} = \sigma^{\boldsymbol{R}}$ for each $\sigma \in \Sigma'$. $\boldsymbol{R}_{\Sigma'}$ is called the $\Sigma'$-*reduct* of $\boldsymbol{R}$. $\Sigma'$ is called a *complete subsignature* of $\Sigma$ if we have equality in both (1) and (2). In this case $\Sigma'$ is completely determined by its sort-set $S'$ and may be denoted by $\Sigma_{S'}$; similarly, the $\Sigma'$-reduct $\boldsymbol{R}_{\Sigma'}$ may be written simply $\boldsymbol{R}_{S'}$.

The set of $\Sigma$-terms over the variables $X$ is denoted by $\mathrm{Te}_\Sigma(X)$ and the corresponding *term* $\Sigma$-*algebra* by $\mathbf{Te}_\Sigma(X)$. Note that $\mathrm{Te}_\Sigma(X) = \mathrm{Te}_{\Sigma_O}(X)$ since only operation symbols of $\Sigma$ are involved in the construction of terms.

Let $S$ be any nonempty set. A mapping $C$ from the power set of $S$ into itself is a *closure operator* if it satisfies the following axioms for all $X, Y \subseteq S$: $X \subseteq Y$ implies $C(X) \subseteq C(Y)$; $X \subseteq C(X)$; $C(C(X)) = C(X)$. A closure operator is *algebraic* if $C(X) = \bigcup \{ Y : Y \subseteq X, Y \text{ finite} \}$. Subsets of $S$ with the property $C(X) = X$ are said to be *closed*. The universal set $S$ is always closed and if $\mathscr{F}$ is any family of closed sets, then $\bigcap \mathscr{F}$ is closed and, if $C$ is algebraic, so is $\bigcup \mathscr{F}$ provided $\mathscr{F}$ is *directed* ($X, Y \in \mathscr{F}$ implies $X \cup Y \subseteq Z$ for some $Z \in \mathscr{F}$). Any family $\mathscr{C}$ of subsets of a nonempty set $S$ satisfying these three properties ($S \in \mathscr{C}$ and $\mathscr{C}$ is closed under arbitrary intersection and directed union) is called an *algebraic closed-set system*. For any algebraic closed-set system $\mathscr{C}$, $C(X) := \bigcap \{ Y : X \subset Y \in \mathscr{C} \}$ for each $X \subseteq S$ defines an algebraic closure operator. If $\mathscr{C}$ is the family of closed sets of an algebraic closure operator, then the algebraic closure operator $C$ obtained from $\mathscr{C}$ in this way coincides with the original.

## 1. Classical data types

In this section we review the algebraization of the classical universal first-order theory of a relational structure as it is developed in [33]. (Only algebraic structures are dealt with in [33], but the methods are easily extended to relational structures.)

It can be viewed as a process by which a relational structure $\boldsymbol{R}$ over an arbitrary many-sorted relational signature is transformed into an algebra $\boldsymbol{A}$ over a new, algebraic signature. The algebraic part of $\boldsymbol{R}$ is a subreduct of its transform. The transformation takes the following form: Let $\Sigma$ be the relational signature of $\boldsymbol{R}$, and let $S$ be its sort-set. The universe of $\boldsymbol{A}$ is obtained by enriching the universe of $\boldsymbol{R}$ by a two-element Boolean algebra

$$\langle \{\mathsf{true}, \mathsf{false}\}, \mathsf{and}, \mathsf{or}, \mathsf{not}, \mathsf{true}, \mathsf{false}\rangle$$

as a new domain of sort bool. An operation $\mathsf{eq}_s^{\boldsymbol{A}}$ of type $ss \to \mathsf{bool}$ is adjoined for each non-Boolean sort $s$: for any $a$, $b$ of type $s$ we have

$$\mathsf{eq}_s^{\boldsymbol{A}}(a, b) = \begin{cases} \mathsf{true} & \text{if } a = b, \\ \mathsf{false} & \text{otherwise.} \end{cases} \tag{1.1}$$

$\mathsf{eq}_s^{\boldsymbol{A}}$ is called the *equality-test operation* over the domain $A_s$ $(= R_s)$.

The nonequality predicates of $\boldsymbol{R}$ are replaced by Boolean-valued operations in a similar way: For each predicate symbol $\sigma$ we have a new operation $\mathsf{pr}_\sigma^{\boldsymbol{A}}$ of type $s_0 \ldots s_{n-1} \to \mathsf{bool}$ where $s_0 \ldots s_{n-1}$ is the type of $\sigma$.

$$\mathsf{pr}_\sigma^{\boldsymbol{A}}(a_0, \ldots, a_{n-1}) = \begin{cases} \mathsf{true} & \text{if } \langle a_0, \ldots, a_{n-1}\rangle \in \sigma^{\boldsymbol{A}}, \\ \mathsf{false} & \text{otherwise.} \end{cases} \tag{1.2}$$

This transforms the relational structure $\boldsymbol{R}$ into an algebra $\boldsymbol{A}$ called the *algebra-transform* of $\boldsymbol{R}$, which we denote by $\mathbf{Alg}\,\boldsymbol{R}$. Its sort-set is $S \cup \{\mathsf{bool}\}$, and its signature is

$$\Sigma(\Lambda) := \Sigma_{\mathrm{O}} \cup \Lambda \cup \{\mathsf{eq}_s : s \in S\} \cup \{\mathsf{pr}_\sigma : \sigma \in \Sigma_{\mathrm{P}}\}, \tag{1.3}$$

where $\Lambda := \{\mathsf{and}, \mathsf{or}, \mathsf{not}, \mathsf{true}, \mathsf{false}\}$ is the signature of Boolean algebras. Its universe $\mathrm{Alg}\,\boldsymbol{R}$ is $R \cup \{\langle \mathsf{bool}, \{\mathsf{true}, \mathsf{false}\}\rangle\}$ where $R = \langle R_s : s \in S\rangle$ is the $S$-sorted universe of $\boldsymbol{R}$. Note that the $\Lambda$-reduct of an arbitrary $\Sigma(\Lambda)$-algebra $\boldsymbol{A}$ is denoted by $\boldsymbol{A}_{\mathsf{bool}}$.

There is no equality-test operation over the Boolean domain, but we may assume one exists without loss of generality. For if we define $\mathsf{eq}_{\mathsf{bool}}$ to be the compound term

$$\mathsf{eq}_{\mathsf{bool}}(x, y) := (\mathsf{not}\ x\ \mathsf{or}\ y)\ \mathsf{and}\ (\mathsf{not}\ y\ \mathsf{or}\ x), \tag{1.4}$$

then (1.1) also holds for $s = \mathsf{bool}$.

The translation of any universal first-order $\Sigma(\Lambda)$-formula $\varphi$ into a $\Sigma(\Lambda)$-term is now obvious. It is defined recursively on the structure of $\varphi$ in the following way, starting with atomic formulas:

$$(t \approx r)^* := \mathsf{eq}_s(t, r),$$

for all $s \in S \cup \{\mathsf{bool}\}$ and $t, r \in (\mathrm{Te}_{\Sigma(\Lambda)}(X))_s$.

$$(\sigma(t_0, \ldots, t_{n-1}))^* := \mathsf{pr}_\sigma(t_0, \ldots, t_{n-1}),$$

for all $\sigma \in \Sigma_{\mathrm{P}}$ and $t_i \in (\mathrm{Te}_\Sigma(X))_{s_i}$, where $s_0, \ldots, s_{n-1}$ is the type of $\sigma$.

$$(\varphi \wedge \psi)^* := \varphi^* \text{ and } \psi^*, \qquad (\varphi \vee \psi)^* := \varphi^* \text{ or } \psi^*,$$
$$(\neg\varphi)^* := \text{not } \varphi^*, \qquad (\varphi \to \psi)^* := \text{not } \varphi^* \text{ or } \psi^*. \tag{1.5}$$

If $\varphi$ is a universal sentence, then $\varphi^* := \psi^*$, where $\psi$ is the quantifier-free part of the prenex normal form of $\varphi$.

Sentences with existential quantifiers in their prenex normal form are harder to handle. One way is to first pass to the Skolem normal form and then apply the above translation. But this entails enriching the signature in a way that depends on the form of the sentence and greatly complicates the algebraization process. We restrict our attention exclusively to universal formulas in this paper.

It now follows almost immediately from the definitions of the notions involved that, for any universal first-order $\Sigma$-sentence $\varphi$ and any relational $\Sigma$-structure $\boldsymbol{R}$,

$$\boldsymbol{R} \models \varphi \text{ iff } \mathbf{Alg}\, \boldsymbol{R} \models \varphi^* \approx \text{true}; \tag{1.6}$$

$\boldsymbol{R} \models \varphi$ means that $\boldsymbol{R}$ is a model of $\varphi$ in the usual sense of first-order predicate logic (see for instance [29]).

Any $\Sigma(\Lambda)$-algebra that is isomorphic to the algebra-transform of some relational $\Sigma$-structure (i.e., that is isomorphic to $\mathbf{Alg}\, \boldsymbol{R}$ for some $\boldsymbol{R}$) will be called an *equality-test algebra over* $\Sigma(\Lambda)$. We give a formal, intrinsic definition for future reference. Any algebraic signature $\Sigma(\Lambda)$ that is constructed from an arbitrary relational signature $\Sigma$ as in (1.3) will be called an *equality-test signature.*

**Definition 1.1.** A $\Sigma(\Lambda)$-algebra $\boldsymbol{A}$ is an *equality-test* (*ET*) *algebra* if the following conditions are satisfied.

(i) $\boldsymbol{A}_{\text{bool}}$ is a two-element Boolean algebra;

(ii) $\text{eq}_s^{\boldsymbol{A}}$ is an equality-test operation on the domain $A_s$ for every $s \in S \backslash \{\text{bool}\}$.

Recall that $\boldsymbol{A}_{\text{bool}} = \boldsymbol{A}_\Lambda$ by convention.

We would have the ideal solution to the problem of algebraizing classical universal first-order logic if we could find a set of equations or conditional equations that characterize the class of ET algebras, but unfortunately the ET algebras do not form a conditional-equational class since they are not closed under direct (i.e., Cartesian) products. But the conditional-equational class generated by the ET algebras is shown in [33] to be axiomatized by the following simple set of conditional equations.

(Axget$_1$) A set of equational axioms for Boolean algebras.

For each $s \in S$:

(Axget$_2$) $\text{eq}_s(x, x) \approx \text{true}$;

(Axget$_3$) $\text{eq}_s(x, y) \leq \text{eq}_s(y, x)$;

(Axget$_4$) $\text{eq}_s(x, y)$ and $\text{eq}_s(y, x) \leq \text{eq}_s(x, z)$.

For each $\sigma \in \Sigma_O$ of type $s_0 \dots s_{n-1} \to s$:

(Axget$_5$) $\mathsf{eq}_{s_0}(x_0, y_0)$ and ... and $\mathsf{eq}_{s_{n-1}}(x_{n-1}, y_{n-1})$
$\leqslant \mathsf{eq}_s(\sigma(x_0, \dots, x_{n-1}), \sigma(y_0, \dots, y_{n-1}))$.

For each $\sigma \in \Sigma_P$ of type $s_0 \dots s_{n-1}$:

(Axget$_6$) $\mathsf{eq}_{s_0}(x_0, y_0)$ and ... and $\mathsf{eq}_{s_{n-1}}(x_{n-1}, y_{n-1})$ and $\sigma(x_0, \dots, x_{n-1})$
$\leqslant \sigma(y_0, \dots, y_{n-1})$;

(Axget$_7$) $\mathsf{eq}_s(x, y) \approx 1 \to x \approx y$.

Note that all of these axioms except the last are equations; $\mathsf{eq}_s(x, y) \leqslant \mathsf{eq}_s(y, x)$ for example can be thought of as an abbreviation of $\mathsf{eq}_s(x, y)$ and $\mathsf{eq}_s(y, x) \approx \mathsf{eq}_s(x, y)$. Let AXGET stand for the set of all these axioms.

The AXGET axioms continue to hold when $s$ is allowed to take the value bool whenever appropriate; recall that $\mathsf{eq}_{\mathrm{bool}}$ is defined in (1.4). Moreover, (Axget$_5$) continues to hold with $\sigma = \mathsf{eq}_s$ for every $s \in S$. In this case the axiom takes the form

$$\mathsf{eq}_s(x_0, y_0) \text{ and } \mathsf{eq}_s(x_1, y_1) \leqslant \mathsf{eq}_{\mathrm{bool}}(\mathsf{eq}_s(x_0, x_1), \mathsf{eq}_s(y_0, y_1)). \tag{1.7}$$

But this follows from (Axget$_3$) and (Axget$_4$), since, in any Boolean algebra, $a$ and $b \leqslant c$ and $a$ and $c \leqslant b$ together imply $a \leqslant$ (not $b$ or $c$) and (not $c$ or $b$).

**Definition 1.2.** A $\Sigma(A)$-algebra satisfying AXGET is called a *generalized equality-test (GET) algebra.*

The main result about equality-test algebras in [33] is the following algebraic representation.

**Theorem 1.3.** *Every GET algebra is isomorphic to a subalgebra of a direct product of ET algebras.*

**Corollary 1.4.** *The class of GET algebras is the smallest conditional-equational class that contains all ET algebras.*

Thus we see AXGET is in a sense the best we can hope for in trying to characterize ET algebras by conditional equations.

The following result is an immediate consequence of Theorem 1.3 together with the equivalence (1.6). For any set $\Gamma$ of universal first-order sentences let $\Gamma^* := \{\varphi^* \approx \mathsf{true}: \varphi \in \Gamma\}$.

**Theorem 1.5.** *Let $\Gamma$ be any set of universal first-order $\Sigma$-sentences. Then $\Gamma^* \cup$ AXGET is a set of conditional-equational axioms for the smallest conditional-equational class that contains* Alg $\boldsymbol{R}$ *for every relational $\Sigma$-structure $\boldsymbol{R}$ such that $\boldsymbol{R} \models \Gamma$.*

Another way of putting this result is that $\Gamma^* \cup$ AXGET is a basis for all conditional equations that hold in the algebra-transform of every model of $\Gamma$.

The equivalence between a universal first-order sentence and its equational transform expressed in (1.6) does not in general hold for GET algebras that are not ET algebras (see [33]).

There are various ways of characterizing ET algebras among the class of GET algebras (although necessarily not by means of conditional equations). An ET algebra may have a proper homomorphic image but every such image must fail to satisfy $(\text{Axget}_7)$ and hence cannot be a GET algebra. (A homomorphic image is *proper* if the corresponding surjective homomorphism is not injective.) Thus every ET algebra is *GET-simple*. The converse follows easily from Theorem 1.3. Hence:

**Theorem 1.6.** *The ET algebras are exactly the GET-simple GET algebras.*

We now describe some consequences of Theorem 1.3 for the specification of ET data types.

By a *data structure* we mean as usual a many-sorted algebra $\boldsymbol{A}$ that is *minimal* in the sense that it has no proper subalgebras; under the assumption that there is at least one ground term of type $s$ for each sort $s$, this is equivalent to the condition that every element of $\boldsymbol{A}$ is the value of at least one ground term. A *data type* is the isomorphism class of a data structure. Let us call a set $\Gamma$ of universal $\Sigma(\Lambda)$-sentences an *initial specification* of a GET data structure $\boldsymbol{A}$ if $\boldsymbol{A}$ is an initial algebra of the conditional-equational class defined by $\Gamma^* \cup \text{AXGET}$. Similarly we call $\Gamma$ a *final specification* of $\boldsymbol{A}$ if $\boldsymbol{A}$ is a final (i.e., terminal) algebra of the class of nontrivial data structures of this conditional-equational class. Suppose $\Gamma$ is an initial specification of an ET data structure $\boldsymbol{A}$. Then every data structure in the class of models of $\Gamma^* \cup \text{AXGET}$ is a homomorphic image of $\boldsymbol{A}$. But since $\boldsymbol{A}$ is an ET algebra, it has no nontrivial GET homomorphic image outside of its data type. Thus the class of ET data structures that are models of $\Gamma^* \cup \text{AXGET}$ must coincide with the data type of $\boldsymbol{A}$, and hence $\Gamma$ is also a final specification of $\boldsymbol{A}$. Conversely, every final specification of an ET data structure must also be an initial specification. Hence any universal specification of an ET data structure is *complete* in the sense that it is both initial and final. One consequence of this fact is that any ET data structure with a finite specification is necessarily computable.

This completes our summary of the main features of the algebraization of the classical universal first-order theory of data structures. The details can be found in [33].

## 2. Nonclassical propositional logics

Before turning to the problem of algebraizing the nonclassical universal first-order theory of data types we take a look at several specific examples of nonclassical propositional logics that can serve as the basis of such a theory; several of them have already occurred in the computer science literature. Each of these logics are

given in the form of a homogeneous algebra, which for reasons to be explained below will be called a *protomatrix*. We will put off until later exactly in what sense one of these protomatrices actually determines a logic.

### 2.1. *Some special logics*

We begin by considering four 3-valued protomatrices. All three have the same signature. It is an extension of the signature of Boolean algebras by a single constant symbol u; in the present context we use the symbols t and f in place of true and false. Although the signature may vary slightly from logic to logic, we will always denote it by $A$. From now on we will refer, on those occasions where it seems appropriate, to the operation symbols of $A$ as *connectives* or *logical connectives*.

- $\mathbf{KW}_3$: *Kleene's weak logic* [20]. See Table 1.
- $\mathbf{KS}_3$: *Kleene's strong logic* [20]. See Table 2. The defining protomatrix coincides with the unique 3-element subdirectly irreducible DeMorgan algebra; see [1].
- $\mathbf{MC}_3$: *McCarthy's noncommuting conditional logic* [27] (see also [17] and [18]). See Table 3.

These three logics are all designed to handle compound statements with atomic components whose truth value may be undetermined either for theoretical reasons (for example if division by zero is involved or an attempt is made to pop an empty stack), or because some computational process fails to terminate. They represent three different ways the logic of such statements can be handled in various actual implementations. If in evaluating the conjunction $R$ and $S$, for instance, both $R$ and $S$ are evaluated independently, before a value for the conjunction is returned,

| and | t | u | f |
|---|---|---|---|
| t | t | u | f |
| u | u | u | u |
| f | f | u | f |

| or | t | u | f |
|---|---|---|---|
| t | t | u | t |
| u | u | u | u |
| f | t | u | f |

| not | |
|---|---|
| t | f |
| u | u |
| f | t |

Table 1.

| and | t | u | f |
|---|---|---|---|
| t | t | u | f |
| u | u | u | f |
| f | f | f | f |

| or | t | u | f |
|---|---|---|---|
| t | t | t | t |
| u | t | u | u |
| f | t | u | f |

| not | |
|---|---|
| t | f |
| u | u |
| f | t |

Table 2.

| and | t | u | f |
|---|---|---|---|
| t | t | u | f |
| u | u | u | u |
| f | f | f | f |

| or | t | u | f |
|---|---|---|---|
| t | t | t | t |
| u | u | u | u |
| f | t | u | f |

| not | |
|---|---|
| t | f |
| u | u |
| f | t |

Table 3.

then either $\mathbf{KW}_3$ or $\mathbf{KS}_3$ would provide the proper evaluation scheme. But if the evaluation of $S$ is delayed until its value is actually required in order to make a decision about the truth value of the conjunction, then $\mathbf{MC}_3$ is the proper choice.

The other 3-valued logic we want to consider is closely related to the logics of Łukasiewicz and Post and can be considered the "classical" logic with three truth values. Since Łukasiewicz's implication is not definable in terms of the other connectives, it is included in the signature.

- $\mathbf{LU}_3$: *Łukasiewicz's 3-valued logic* [24, 35]. See Table 4. The defining protomatrix is *primal* in the sense that every operation of finite rank over the universe is defined by some $\Lambda$-term.

The implication $\rightarrow$ and the negation not are the basic connectives here; conjunction and disjunction are defined in their terms by $x$ and $y :=$ not $(x \rightarrow$ not $y)$ and $x$ or $y :=$ not $x \rightarrow y$.

The next logic has four truth values. It has the same connectives as the classical propositional logic.

- $\mathbf{BE}_4$: *Belnap's 4-valued logic* [2, 3]. See Table 5. The defining protomatrix is the unique 4-element subdirectly irreducible DeMorgan algebra.

This logic is designed to represent our state of knowledge of a database at a particular time rather than the actual state of affairs. We may have been told that a particular ground clause is true or we might have been told it is false; this would result in the clause having one of the classical truth values. But we may also have been given no information about the clause (neither true or false) or we may have been told (presumably by different sources) that it is both true and false. From the fact we have been told only that $R$ is true we may infer only that $R$ or $S$ is true regardless of what we have been told about $S$. But if we have been told only that $R$ is false, then whatever we have been told about $S$ we may also infer about $R$ or $S$. This is the reasoning behind the first and last rows of the or-table. The other entries in all three tables can be explained in a similar way.

The last of the special logics is infinite valued and closely related to intuitionistic logic. The connectives are and, or, $\rightarrow$, and the constants, 0, 1, t.

| $\rightarrow$ | t | u | f |
|---|---|---|---|
| t | t | u | f |
| u | t | t | u |
| f | t | | t |

| and | t | u | f |
|---|---|---|---|
| t | t | u | f |
| u | u | f | f |
| f | f | f | f |

| or | t | u | f |
|---|---|---|---|
| t | t | t | t |
| u | t | t | u |
| f | t | u | f |

| not | |
|---|---|
| t | f |
| u | u |
| f | t |

Table 4.

| and | t | b | n | f |
|---|---|---|---|---|
| t | t | b | n | f |
| b | b | b | f | f |
| n | n | f | n | f |
| f | f | f | f | f |

| or | t | b | n | f |
|---|---|---|---|---|
| t | t | t | t | t |
| b | t | b | t | b |
| n | t | t | n | n |
| f | t | b | n | f |

| not | |
|---|---|
| t | f |
| b | b |
| n | n |
| f | t |

Table 5.

- $\mathbf{HE}_{\omega+1}$: *$(\omega+1)$-valued intuitionistic logic.* Let $\mathbb{N}_{\omega+1} := \{0, 1, 2, \ldots, \mathrm{t}\}$ and

$$\mathbf{HE}_{\omega+1} := \langle \mathbb{N}_{\omega+1}, \text{and}, \text{or}, \rightarrow, \mathrm{t} \rangle,$$

where the operations are defined as follows: $x$ and $y := \min\{x, y\}$ (with respect to the natural ordering with t as largest element); $x$ or $y := \max\{x, y\}$;

$$x \rightarrow y := \begin{cases} \mathrm{t} & \text{if } x \leqslant y, \\ y & \text{otherwise.} \end{cases}$$

$\mathbf{HE}_{\omega+1}$ is a special kind of Heyting algebra. See [19]. For general information about Heyting algebras see [1] or [34].

The use of $\mathbf{HE}_{\omega+1}$ as the underlying logic of a data structure is closely related to the use of complete partial orderings as the model for the semantics of programming languages; see for instance [36]. Assume that the structure of data objects can in general be completely determined only on the basis of an infinite computational process, but that every object $d$ is the limit in a natural way of finitely computed subobjects $d_0, d_1, d_2, \ldots, d_n, \ldots$, which approximate $d$ with increasing accuracy. (For example, $d$ could be a computable function with domain $\mathbb{N} := \{0, 1, \ldots\}$ and $d_n$ the restriction of $d$ to $\{0, 1, \ldots, n-1\}$. We can then assign to each atomic formula a unique ordinal $n \in \mathbb{N}_{\omega+1}$ in a natural way. The ordinal assigned to an equality $d \approx e$ between two data objects would be the least upper bound of all $n$ such that $d_n = e_n$. Suppose $\varphi(d, e, f, \ldots)$ is a compound expression involving the connectives and, or, $\rightarrow$, and the data objects $d, e, f, \ldots$ It is not difficult to see that the ordinal assigned to $\varphi$ on the basis of the definitions of and, or, and $\rightarrow$ given for $\mathbf{HE}_{\omega+1}$ is the largest $n$ such that $\varphi(d_n, e_n, f_n, \ldots)$ is true in the classical sense.

Reference [12] is an early paper on the special kind of logic we have been discussing. Ginsberg [11] discusses the role of multiple-valued logics in computer science and artificial intelligence in general. For some work on the role of Kleene's strong logic in logic programming see [32, 22]. For other applications of multiple-valued logic to logic programming see [10, 21].

### 2.2. *General nonclassical logics*

In what sense do the above protomatrices determine a logic? In fact several different logics can be associated with each of them, and it is not always clear which one is intended in a particular circumstance. It is obvious that every algebra has an associated *equational* logic; the fundamental propositional form here is an equation between two terms over the signature of the algebra. But each protomatrix also gives rise to several *assertional* logics where the fundamental propositional forms are the terms themselves.

For example the 2-element Boolean algebra $\boldsymbol{B}_2$ as a protomatrix determines two assertional logics whose asserted propositions are respectively the tautologies and the contradictions. It does not matter which of the three one chooses in this case since they are all equivalent in a natural sense. (An equation $\varphi \approx \psi$ is universally

satisfied in $\boldsymbol{B}_2$ iff $\varphi \leftrightarrow \psi$ is a tautology iff not$(\varphi \leftrightarrow \psi)$ is a contradiction, and conversely $\varphi$ is a tautology iff not $\varphi$ is a contradiction iff $\varphi \approx$ true is universally satisfied.) Consider now the protomatrix $\mathbf{MC}_3$ of McCarthy's noncommuting conditional logic. Gries seems clearly to have an assertional logic in mind when he deals with this protomatrix in [17]. On the other hand, in their detailed study of the logic of $\mathbf{MC}_3$ carried out in [18], Guzman and Squires deal exclusively with the equational logic. The exact connection between the assertional and the equational logic in this case is not clear, but it is certainly not as strong as in the classical (Boolean) case. For an arbitrary protomatrix the connection between the equational and assertional logics may be tenuous. But the algebraic theory of data types seems to rely heavily on the assumption of a close connection between the two logics. In this paper we will consider a large class of protomatrices for which the connection is nearly as strong as it is in the classical case, and for these protomatrices the algebraization of the associated universal first-order logic can be carried out almost as smoothly. For protomatrices that are not contained in this class the situation is still problematical.

We begin our investigation of nonclassical algebraic theory of data types by taking a more detailed look at assertional propositional logics in general. The notion we have in mind is the one due essentially to Tarski [37], where an assertional logic is characterized as a formally defined set of formulas together with a relation of *entailment* or *consequence*. The formulas can be identified with the terms of a homogeneous signature $\Lambda$. The operation symbols $\gamma \in \Lambda$ are to be viewed, in this context, as propositional connectives; the variables $x, x_0, x_1, \ldots, y, y_0, \ldots, z, \ldots$ as propositional variables; and the terms $t \in \mathrm{Te}_\Lambda(X)$ as propositional formulas. Formally an *assertional* or *propositional logic*, or simply a *logic*, is an ordered pair $\mathscr{L} := \langle \Lambda, \vdash_{\mathscr{L}} \rangle$ where $\vdash_{\mathscr{L}}$ is an abstract relation between a set $T$ of terms and an individual term $r$ that satisfies several well known axioms (see for instance [4]). What these axioms say in effect is that the mapping

$$T \mapsto \{r\colon T \vdash_{\mathscr{L}} r\} \tag{2.1}$$

defines an algebraic closure operation on $\mathrm{Te}_\Lambda(X)$ with the additional property that $T \vdash_{\mathscr{L}} r$ implies $\{h(t)\colon t \in T\} \vdash_{\mathscr{L}} h(r)$ for every substitution $h$ of terms for variables. We say that $T$ *entails* $r$ and that $r$ is a *consequence of* or *deducible from* $T$ in $\mathscr{L}$ whenever $T \vdash_{\mathscr{L}} r$. The algebraic closure operator on sets of terms defined in (2.1) is the *consequence operator* associated with $\mathscr{L}$. Closed sets of terms are called *$\mathscr{L}$-theories*, or simply *theories* when the logic is clear. The *theorems* of $\mathscr{L}$ are the terms $r$ such that $\vdash_{\mathscr{L}} r$, i.e., that are deducible from the empty set of terms. (The theorems constitute the smallest closed set of terms.) The *classical propositional logic* $\mathscr{CPL}$ can be characterized in the following way: $\mathscr{CPL} := \langle \Lambda, \vdash_{\mathscr{CPL}} \rangle$ where $\Lambda := \{$and, or, not, true, false$\}$ and $t_0, \ldots, t_{n-1} \vdash_{\mathscr{CPL}} r$ iff not$(t_0$ and $\cdots$ and $t_{n-1})$ or $r$ is a tautology.

Assertional logics are usually defined axiomatically by specifying certain terms in $\mathrm{Te}_\Lambda(X)$ as (*logical*) *axioms* (actually *axiom schemata*) and certain *rules of inference*

of the form

$$\frac{t_0, \ldots, t_{n-1}}{r}.$$

A *derivation* of a term $r$ from a set of terms $T$ is a sequence of terms with the following property: each member is either an element of $T$, a substitution instance of an axiom, or a substitution instance of the conclusion $r$ of an inference rule such that the corresponding substitution instances of the premises $t_0, \ldots, t_{n-1}$ occur earlier in the sequence. Then $T \vdash_{\mathscr{L}} r$ iff there is a derivation of $r$ from $T$. Every assertional logic can be defined axiomatically in this way ([23]).

It is well known (see [39, 40]) that every assertional logic $\mathscr{L}$ in the above sense can also be characterized by means of matrices. By a *matrix over* $\Lambda$, or a *$\Lambda$-matrix*, we mean a pair $M := \langle A, D\rangle$ where $A$ is a $\Lambda$-algebra and $D$ is an arbitrary subset of the universe $A$ of $A$. The elements of $D$ are called the *designated truth values* of $M$, and the underlying algebra $A$ is called a *protomatrix*.

Let $t(x_0, \ldots, x_{n-1}) \in \mathrm{Te}_\Lambda(X)$, where $x_0, \ldots, x_{n-1}$ is a list of variables occurring in $t$ (and possibly some additional ones). For any $\Lambda$-algebra $A$ and $a_0, \ldots, a_{n-1} \in A$, we denote by $t^A(a_0, \ldots, a_{n-1})$ the value $t$ takes when $x_0, \ldots, x_{n-1}$ are assigned the values $a_0, \ldots, a_{n-1}$, respectively. (We often write just $\bar{x}$, $\bar{a}$, $t(\bar{x})$, and $t^A(\bar{a})$.) Every class $\mathbf{M}$ of matrices defines a unique assertional logic in the following way:

$$t_0(\bar{x}), \ldots, t_{m-1}(\bar{x}) \models_{\mathbf{M}} r(\bar{x}) \text{ iff } t_0^A(\bar{a}), \ldots, t_{m-1}^A(\bar{a}) \in D \text{ implies } r^A(\bar{a}) \in D \text{ for every } \langle A, D\rangle \in \mathbf{M} \text{ and every assignment } \bar{a} \text{ in } A; \tag{2.2}$$

or in more algebraic terms, $h(t_0), \ldots, h(t_{m-1}) \in D$ implies $h(t) \in D$, for every homomorphism $h : \mathrm{Te}_\Lambda(X) \to A$. Then $\langle \Lambda, \models_{\mathbf{M}}\rangle$ is a logic and, conversely, for every logic $\mathscr{L} := \langle \Lambda, \vdash_{\mathscr{L}}\rangle$ there is a set of matrices $\mathbf{M}$ such that $\models_{\mathbf{M}}$ and $\vdash_{\mathscr{L}}$ coincide. The classical propositional logic can be defined in this way by the matrix $\langle \boldsymbol{B}_2, \{\text{true}\}\rangle$ where $\boldsymbol{B}_2$ is the 2-element Boolean algebra; thus $\vdash_{\mathscr{CPL}}$ and $\models_{\boldsymbol{B}_2}$ coincide. (When no confusion is likely we do not bother to distinguish between a matrix and its protomatrix.)

Let $\mathscr{L} := \langle \Lambda, \vdash_{\mathscr{L}}\rangle$ be an arbitrary logic and $A$ an arbitrary $\Lambda$-algebra. A nonempty subset $F$ of $A$ is called an *$\mathscr{L}$-filter* of $A$ if

$$t_0(\bar{x}), \ldots, t_{m-1}(\bar{x}) \vdash_{\mathscr{L}} r(\bar{x})$$

implies

$$t_0^A(\bar{a}), \ldots, t_{m-1}^A(\bar{a}) \in F \text{ implies } r(\bar{a}) \in F$$

for all $t_0, \ldots, t_{m-1}, r \in \mathrm{Te}_\Lambda(X)$ and all assignments $\bar{a}$. Let $\mathrm{Fi}_{\mathscr{L}}A$ be the set of all $\mathscr{L}$-filters of $A$. $\mathrm{Fi}_{\mathscr{L}}A$ is an algebraic closure system over $A$ (see the Introduction). For any subset $B$ of $A$, we denote by $[B]_{\mathscr{L}}$, or just $[B]$ when $\mathscr{L}$ is clear from context, the smallest $\mathscr{L}$-filter that includes $B$; for any filter $F$, $F[B]$ is the smallest filter that includes $F \cup B$. If $A$ is a Boolean algebra, then its $\mathscr{CPL}$-filters are just the Boolean filters in the usual sense (i.e., the subsets $F$ of $A$ such that $\text{true} \in F$, $b \in F$ whenever $a \in F$ and $a \leq b$, and $a$ and $b \in F$ whenever $a, b \in F$).

Note that the filters on the algebra of terms $\mathbf{Te}_\Lambda(X)$ are just the *$\mathscr{L}$-theories.* Consequence can be expressed in terms of theories as follows:

$$t_0, \ldots, t_{m-1} \vdash_{\mathscr{L}} r \text{ iff } r \in [t_0, \ldots, t_{m-1}]_{\mathscr{L}}. \tag{2.3}$$

We say that the *deduction theorem holds for* a logic $\mathscr{L}$ if there exists a term $\mathsf{de} \in \mathrm{Te}_\Lambda(\{x, y\})$ such that

$$t_0, \ldots, t_{m-1}, r \vdash_{\mathscr{L}} s \text{ iff } t_0, \ldots, t_{m-1} \vdash_{\mathscr{L}} \mathsf{de}(r, s)$$

for all $t_0, \ldots, t_{n-1}, r, s \in \mathrm{Te}_\Lambda(X)$. The term de is called a *deduction term* for $\mathscr{L}$. The deduction theorem holds for $\mathscr{CPL}$ with deduction term $\mathsf{de}(x, y) := \text{not } x \text{ or } y$.

The following proposition is an immediate consequence of the characterization of the consequence relation in terms of theories.

**Proposition 2.1.** *Assume the deduction theorem holds for $\mathscr{L}$ with deduction term* de. *For any $\mathscr{L}$-theory and all $t, r \in \mathrm{Te}_\Lambda(X)$ we have*

$$r \in T[t] \text{ iff } \mathsf{de}(t, r) \in T.$$

For a systematic discussion of the modern theory of propositional logics see [7, 40]. For a different view of general logic with a special emphasis on applications to computer science see [30].

### *2.3. Standard logics*

For the purposes of this paper we will restrict our attention almost exclusively to assertional logics that are defined by a set of matrices of the form $\langle \boldsymbol{L}, \{\mathrm{t}\}\rangle$ where $\boldsymbol{L}$ is an arbitrary (possibly infinite) $\Lambda$-algebra and t is a constant symbol of $\Lambda$. In addition we assume that every element of $\boldsymbol{L}$ is denoted by a ground term, i.e., $\boldsymbol{L}$ is a data structure. We will call such matrices *standard*, and a *standard logic* will be one of the form $\langle \Lambda, \models_{\mathbf{L}}\rangle$ where $\mathbf{L}$ is a class of standard matrices. (The term *standard logic* has a different meaning in the literature of propositional logic.) A standard matrix is completely determined by its protomatrix.

We can associate a natural standard logic with each of the special protomatrices $\boldsymbol{A}$ mentioned in the first part of the section by taking $\mathbf{L} := \{\langle \boldsymbol{A}, \{\mathrm{t}\}\rangle\}$, but there are other, possibly more natural, logics associated with them. One such logic is considered by Belnap in [2]. The protomatrix $\mathbf{BE}_4$ with or as join and and as meet forms a lattice called the *logic lattice* in [2]; its Hasse diagram is given in Fig. 1.

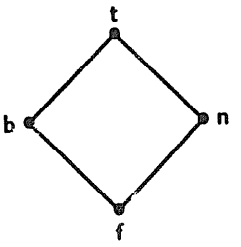


Fig. 1.

Belnap defines a logic $\langle \Lambda, \models \rangle$ in terms of the ordering relation of this lattice:

$$t_0(\bar{x}), \ldots, t_{m-1}(\bar{x}) \models r(\bar{x}) \text{ iff } t_0^{\mathbf{BE}_4}(\bar{a}), \ldots, t_{m-1}^{\mathbf{BE}_4}(\bar{a}) \leqslant r^{\mathbf{BE}_4}(\bar{a})$$

for all assignments $\bar{a}$ in $\mathbf{BE}_4$.

This is the same logic that is defined in our sense by taking $\mathbf{L}$ to consist of the three matrices $\langle \mathbf{BE}_4, \{t\} \rangle$, $\langle \mathbf{BE}_4, \{b, t\} \rangle$, and $\langle \mathbf{BE}_4, \{n, t\} \rangle$. We will refer to this logic in the sequel as the *strong 4-valued Belnap logic.* It is not standard.

The logic defined by $\langle \mathbf{HE}_{\omega+1}, \{t\} \rangle$ is called *linear intuitionistic logic* ($\mathscr{LJL}$). It can be axiomatized by adding the single axiom $(x \to y)$ or $(y \to x)$ to the axioms of intuitionistic logic. It can also be defined by a finite set of finite standard matrices. See [8].

The is well known that the deduction theorem holds for $\mathscr{LJL}$ with $\mathsf{de}(x, y) := x \to y$ as the deduction term. The deduction theorem also holds for 3-valued Łukasiewicz logic defined by the standard matrix $\langle \mathbf{LU}_3, \{t\} \rangle$. In this case the deduction term is $\mathsf{de}(x, y) := x \to (x \to y)$.

If the matrix $\langle \boldsymbol{L}, \{t\} \rangle$ is one of the members of the defining set $\mathbf{L}$ of a standard logic $\mathscr{L}$, then $\{t\}$ will be an $\mathscr{L}$-filter and must be included in every other $\mathscr{L}$-filter of $\boldsymbol{L}$. Thus $\mathrm{Fi}_{\mathscr{L}}\boldsymbol{L}$ always contains at least two $\mathscr{L}$-filters, $\{t\}$ and $\boldsymbol{L}$. It may of course contain others.

We mention one other important property of standard logics. Every relation of consequence $t_0, \ldots, t_{n-1} \vdash_{\mathscr{L}} r$ is equivalent to the conditional equation

$$t_0 \approx t \wedge \cdots \wedge t_{n-1} \approx t \to r \approx t$$

holding identically in each member of the set of protomatrices that define $\mathscr{L}$.

## 3. Equality-test algebras over nonclassical logics

Let $\mathscr{L} := \langle \Lambda, \vdash_{\mathscr{L}} \rangle$ an arbitrary standard assertional logic that is defined by a set $\mathbf{L}$ of protomatrices (i.e., matrices in standard form). $\mathscr{L}$ and $\mathbf{L}$ are assumed fixed throughout the section. We describe the *universal first-order logic over* $\mathscr{L}$, actually one such logic for each relational signature. Let $\Sigma$ be such a signature and $S$ its sort-set. The universal first-order $\Sigma$-language over $\mathscr{L}$ is defined like the corresponding classical first-order $\Sigma$-language except that the classical propositional connectives are replaced by the special connectives of $\Lambda$. We call the (quantifier-free) formulas of this language $(\Sigma, \Lambda)$-*formulas.* The *universal* $(\Sigma, \Lambda)$-*sentences* are the universal closures of the $(\Sigma, \Lambda)$-formulas, but we normally omit the quantifiers thus, in effect, identifying formulas and universal sentences. Recall that the sets of operation and predicate symbols are denoted by $\Sigma_{\mathrm{O}}$ and $\Sigma_{\mathrm{P}}$, respectively.

### 3.1. Structures over nonclassical logics

An *admissible relational* $\Sigma$-*structure* $\boldsymbol{R}$ *over* $\mathscr{L}$ consists of an $S$-sorted universe $R := \langle R_s : s \in S \rangle$ and interpretations of the operation and predicate symbols of $\Sigma$. If

$\sigma \in \Sigma_O$, its interpretation $\sigma^R$ is the same as in the classical case: $\sigma^R : R_{s_0} \times \cdots \times R_{s_{n-1}} \to R_s$ where $s_0 \ldots s_{n-1} \to s$ is the type of $\sigma$. (So $\langle R, \sigma^R \rangle_{\sigma \in \Sigma_O}$ is a $\Sigma_O$-algebra in the usual sense.) Predicate symbols different from the equality symbol are also interpreted in the classical way, except that the two classical truth values are replaced by the elements of $L$, the universe of one of the protomatrices in the defining set $\mathbf{L}$ of $\mathscr{L}$. Thus if $\sigma$ is a predicate symbol of type $s_0, \ldots, s_{n-1}$, $\sigma^R : R_{s_0} \times \cdots \times R_{s_{n-1}} \to L$. The interpretation $\approx_s^R$ of the equality symbol of type $s$ is a mapping of $R_s \times R_s$ into $L$. In contrast to the classical case, the fundamental properties of equality over a general nonclassical logic are not entirely obvious. The problem of identifying them has not yet been completely resolved, but finding the right way to interpret equality is crucial to obtaining a satisfactory algebraization of the universal first-order logic over $\mathscr{L}$. We hope the definition we now give will be justified by the subsequent development. We normally write $a \approx_s^R b$ instead of $\approx_s^R(a, b)$.

**Definition 3.1.** A family of mappings $\approx_s^R : R_s \times R_s \to L$, for $s \in S$, is called an *$\mathscr{L}$-admissible interpretation of equality* if it satisfies the following conditions:

(i) $a \approx_s^R b = \mathrm{t}$ iff $a = b$.

For every $\mathscr{L}$-filter $F$ of $L$:

(ii) $a \approx_s^R b \in F$ implies $b \approx_s^R a \in F$;

(iii) $a \approx_s^R b \in F$ and $b \approx_s^R c \in F$ implies $a \approx_s^R c \in F$.

(iv) For each $\sigma \in \Sigma_O$ of type $s_0 \ldots s_{n-1} \to s$:

$$a_0 \approx_{s_0}^R b_0, \ldots, a_{n-1} \approx_{s_{n-1}}^R b_{n-1} \in F$$

implies $\sigma^R(a_0, \ldots, a_{n-1}) \approx_s^R \sigma^R(b_0, \ldots, b_{n-1}) \in F$.

(v) For each $\sigma \in \Sigma_P$ of type $s_0 \ldots s_{n-1}$:

$$a_0 \approx_{s_0}^R b_0, \ldots, a_{n-1} \approx_{s_{n-1}}^R b_{n-1}, \sigma^R(a_0, \ldots, a_{n-1}) \in F$$

implies $\sigma^R(b_0, \ldots, b_{n-1}) \in F$.

Note that in the event $\{\mathrm{t}\}$ and $L$ are the only $\mathscr{L}$-filters of $L$, conditions (ii)-(v) are consequences of condition (i). This is the situation in classical logic where $L$ is the 2-element Boolean algebra $\mathbf{B}_2$. Moreover, in this case condition (i) clearly reduces to the standard interpretation of equality as identity:

$$a \approx_s^R b = \begin{cases} \text{true} & \text{if } a = b, \\ \text{false} & \text{otherwise.} \end{cases}$$

The notion of an *admissible relational $\Sigma$-structure over $\mathscr{L}$* is now completely described. In summary it consists of an $S$-sorted universe and interpretations of all operation and predicate symbols of $\Sigma$, with the interpretations of the predicate symbols, including equality, taking truth values in one of the defining protomatrices $L$ of $\mathscr{L}$. In addition the interpretation of equality must be $\mathscr{L}$-admissible. Observe that a $\Sigma$-structure in the classical sense is just an admissible $\Sigma$-structure over $\mathbf{B}_2$.

The essential property of an admissible relational $\Sigma$-structure over $\mathscr{L}$ is the $\mathscr{L}$-admissibility of the equality interpretation. The classical equality-test algebras have the property, although in a trivial sense because $\boldsymbol{B}_2$ has no proper nontriviai filters. More significant is the fact that every generalized equality-test algebra has the property. Indeed $\mathscr{CPL}$-admissibility is the essential part of the definition of GET algebras in Definition 1.2 that allows us to obtain the representation Theorem 1.3. The most natural data structures over the nonclassical logic $\mathscr{LIL}$ also have an admissible equality interpretation. Consider for example the set $R$ of all functions on the positive integers. For all $f, g \in R$ let $f \approx^{R} g$ be the greatest $n \in \mathbb{N}_{\omega+1}$ such that $f(k) = g(k)$ for all $k < n$. Then $\approx^{R}$ is $\mathscr{LIL}$-admissible.

### 3.2. *The algebra-transform*

With the classical case in mind it is obvious how to define the *algebra-transform* **Alg** $\boldsymbol{R}$ of an admissible relational $\Sigma$-structure $\boldsymbol{R}$ over the logic $\mathscr{L}$. Its sort-set is $S \cup \{\log\}$, and its signature $\Sigma(\Lambda)$ is defined just as in the classical case (see (1.3)); of course we now replace the sort bool by log and take $\Lambda$ to be the signature of $\mathscr{L}$. Note that the $\Lambda$-reduct of an arbitrary $\Sigma(\Lambda)$-algebra $\boldsymbol{A}$ is denoted by $\boldsymbol{A}_{\log}$. Formally **Alg** $\boldsymbol{R}$ is defined by the following conditions (in which we write $\boldsymbol{A}$ for **Alg** $\boldsymbol{R}$):

$$\boldsymbol{A}_{\Sigma_O} := \boldsymbol{R}_{\Sigma_O},$$

$$\boldsymbol{A}_{\log} := \boldsymbol{L},$$

$$\mathrm{pr}_{\sigma}^{\boldsymbol{A}} := \sigma^{\boldsymbol{R}} \quad \text{for each } \sigma \in \Sigma_P,$$

$$\mathrm{eq}_{s}^{\boldsymbol{A}} := \approx_{s}^{\boldsymbol{R}} \quad \text{for each } s \in S.$$

The translation of a universal $\Sigma(\Lambda)$-sentence $\varphi$ into a $\Sigma(\Lambda)$-term $\varphi^*$ is defined as in the classical case with the obvious modifications, the main one being to replace the equalities (1.5) by

$$(\gamma(\varphi_0, \ldots, \varphi_{n-1}))^* := \gamma(\varphi_0^*, \ldots, \varphi_{n-1}^*), \quad \gamma \in \Lambda.$$

The notion of an admissible relation $\Sigma$-structure $\boldsymbol{R}$ over $\mathscr{L}$ being a *model* of a universal $(\Sigma, \Lambda)$-sentence $\varphi$, in symbols $\boldsymbol{R} \models_{\mathscr{L}} \varphi$, can now be defined by the equivalence

$$\boldsymbol{R} \models_{\mathscr{L}} \varphi \text{ iff } \mathbf{Alg}\, \boldsymbol{R} \models \varphi^* \approx \mathrm{t}. \tag{3.1}$$

Continuing the analogy with classical logic we call an algebraic signature of the form $\Sigma(\Lambda)$ an *equality-test signature over* $\mathscr{L}$, and any algebra isomorphic to an algebra of the form **Alg** $\boldsymbol{R}$ an *admissible equality-test algebra over* $\mathscr{L}$. We give a formal, intrinsic definition in Definition 3.3. We first define the notion of an $\mathscr{L}$-admissible family of equality-test operations. It is convenient for future purposes to define the notion for an arbitrary $\Sigma(\Lambda)$-algebra $\boldsymbol{A}$ whose $\Lambda$-reduct $\boldsymbol{A}_{\log}$ can be

any $\Lambda$-algebra and not just one of the defining protomatrices of $\mathscr{L}$. Recall that a nonempty subset $F$ of $A$ is called an *$\mathscr{L}$-filter* if $t_0(\bar{x}), \ldots, t_{m-1}(\bar{x}) \vdash_{\mathscr{L}} r(\bar{x})$ implies

$$t_0^{\mathbf{A}}(\bar{a}), \ldots, t_{m-1}^{\mathbf{A}}(\bar{a}) \in F \text{ implies } r(\bar{a}) \in F$$

for all $t_0, \ldots, t_{m-1}, r \in \mathrm{Te}_{\Lambda}(X)$ and all assignments $\bar{a}$.

**Definition 3.2.** Let $\Sigma(\Lambda)$ be an equality-test signature over $\mathscr{L}$ and let $\mathbf{A}$ be an arbitrary $\Sigma(\Lambda)$-algebra. The family of operations $\{\mathrm{eq}_s^{\mathbf{A}} : s \in S\}$ is called a *$\mathscr{L}$-admissible equality-test system* on $\mathbf{A}$ if the following conditions hold for all $s \in S$:

(i) $\mathrm{eq}_s^{\mathbf{A}}(a, b) = \mathrm{t}$ iff $a = b$.

For every $\mathscr{L}$-filter $F$ of $\mathbf{A}_{\log}$:

(ii) $\mathrm{eq}_s^{\mathbf{A}}(a, b) \in F$ implies $\mathrm{eq}_s^{\mathbf{A}}(b, a) \in F$;

(iii) $\mathrm{eq}_s^{\mathbf{A}}(a, b) \in F$ and $\mathrm{eq}_s^{\mathbf{A}}(b, c) \in F$ implies $\mathrm{eq}_s^{\mathbf{A}}(a, c) \in F$.

(iv) For all $\sigma \in \Sigma_{\mathrm{O}}$ of type $s_0 \ldots s_{n-1} \to s$:

$$\mathrm{eq}_{s_0}^{\mathbf{A}}(a_0, b_0), \ldots, \mathrm{eq}_{s_{n-1}}^{\mathbf{A}}(a_{n-1}, b_{n-1}) \in F$$

$$\text{implies } \mathrm{eq}_s^{\mathbf{A}}(\sigma^{\mathbf{R}}(a_0, \ldots, a_{n-1}), \sigma^{\mathbf{R}}(b_0, \ldots, b_{n-1})) \in F.$$

(v) For all $\sigma \in \Sigma_{\mathrm{P}}$ of type $s_0 \ldots s_{n-1}$:

$$\mathrm{eq}_{s_0}^{\mathbf{A}}(a_0, b_0), \ldots, \mathrm{eq}_{s_{n-1}}^{\mathbf{A}}(a_{n-1}, b_{n-1}), \sigma^{\mathbf{A}}(a_0, \ldots, a_{n-1}) \in F$$

$$\text{implies } \sigma^{\mathbf{A}}(b_0, \ldots, b_{n-1}) \in F.$$

Compare the following definition with Definitions 1.1 and 3.1.

**Definition 3.3.** Let $\mathscr{L}$ be a standard logic defined by a set $\mathbf{L}$ of standard matrices. A $\Sigma(\Lambda)$-algebra $\mathbf{A}$ is an *admissible equality-test algebra* over $\mathscr{L}$ (an $\mathrm{ET}_{\mathscr{L}}$ *algebra*) if the following conditions are satisfied:

(i) $\mathbf{A}_{\log} \in \mathbf{L}$;

(ii) $\{\mathrm{eq}_s^{\mathbf{A}} : s \in S\}$ is an $\mathscr{L}$-admissible equality-test system on $\mathbf{A}$.

Extending the analogy with classical equality-test algebras we want to consider the possibility that there exists an equality-test operation $\mathrm{eq}_{\log}$ on the logic-domain. This might be introduced in the form of new operation symbol, or, as in the classical case, it may already exist in the form of a term built up from the logical connectives in $\Lambda$. An $\mathrm{ET}_{\mathscr{L}}$ algebra is called *special* if it has an additional operation $\mathrm{eq}_{\log}$ of type $\log \log \to \log$ satisfying conditions (i)–(iv) of Definition 3.2 with $s_0, \ldots, s_{n-1}, s = \log$ and $\sigma \in \Lambda$. In addition it is also required that the following condition hold for each $s \in S$ and every $\mathscr{L}$-filter $F$ of $\mathbf{A}_{\log}$.

(iii) $\mathrm{eq}_s^{\mathbf{A}}(a_0, b_0), \mathrm{eq}_s^{\mathbf{A}}(a_1, b_1) \in F$ implies $\mathrm{eq}_{\log}^{\mathbf{A}}(\mathrm{eq}_s^{\mathbf{A}}(a_0, a_1), \mathrm{eq}_s^{\mathbf{A}}(b_0, b_1)) \in F$.

Every classical ET algebra is special; see (1.7).

An admissible relational $\Sigma$-structure $\mathbf{R}$ over $\mathscr{L}$ is to *special* if its corresponding algebra-transform $\mathbf{Alg}\,\mathbf{R}$ is a special $\mathrm{ET}_{\mathscr{L}}$ algebra.

An equality-test algebra can be viewed as a data structure that incorporates a part of its underlying logic within its structure. We take a universal algebra approach in this paper, but category theory can also be used. For some work along this line see [6].

## 4. Autoalgebraizable logics

Admissible equality-test algebras over a nonclassical logic $\mathscr{L}$ may be viewed as the natural analogue of the classical, or Boolean, equality-test algebras. As in the Boolean case they cannot be completely characterized by conditional equations, and in this section and the next we investigate just how well they can be described by this means. We will see that for a wide class of logics, the so-called *autoalgebraizable* logics for which the deduction theorem holds, the situation is almost as good as in the Boolean case.

Throughout this section, unless otherwise noted, $\mathscr{L} = \langle \Lambda, \vdash_{\mathscr{L}} \rangle$ will be a standard logic defined by a set $\mathbf{L}$ of matrices of the form $\langle \boldsymbol{L}, \{t\} \rangle$ where $\boldsymbol{L}$ is a data structure; as usual we often fail to distinguish between a matrix and its protomatrix. We focus our attention exclusively on the logic domains of what will eventually turn out to be the $\Lambda$-reducts of generalized admissible equality-test algebras over $\mathscr{L}$. Consequently we will be dealing exclusively with algebras of the homogeneous signature $\Lambda$. In particular, $\boldsymbol{A}$ will always be a $\Lambda$-algebra.

In a systematic study of admissible equality-test systems it proves convenient to separate the notion of admissibility from that of an equality-test. By a *binary operation* on a set $A$ we mean any function $f: A \times A \to A$; if $A$ is the carrier of an algebra $\boldsymbol{A}$, then $f$ is said to be *term-definable* if there is a $t \in \mathrm{Te}_{\Lambda}(x, y)$ such that $f(a, b) = t^{\boldsymbol{A}}(a, b)$ for all $a, b \in A$.

**Definition 4.1.** Let $\mathscr{L}$ be a logic (not necessarily standard) and $\boldsymbol{A}$ a $\Lambda$-algebra. A binary operation $\mathrm{eq}: A \times A \to A$ is said to be *$\mathscr{L}$-admissible* for $\boldsymbol{A}$ if the following conditions hold for every $\mathscr{L}$-filter $F$ of $\boldsymbol{A}$ and all $a, b, c, a_0, \ldots, b_0, \ldots, b_{n-1} \in A$.

(i) $\mathrm{eq}(a, a) \in F$;

(ii) $\mathrm{eq}(a, b) \in F$ implies $\mathrm{eq}(b, a) \in F$;

(iii) $\mathrm{eq}(a, b), \mathrm{eq}(b, c) \in F$ implies $\mathrm{eq}(a, c) \in F$;

(iv) for all $\gamma \in \Lambda$,

$$\mathrm{eq}(a_0, b_0), \ldots, \mathrm{eq}(a_{n-1}, b_{n-1}) \in F$$
$$\text{implies } \mathrm{eq}(\gamma^{\boldsymbol{A}}(a_0, \ldots, a_{n-1}), \gamma^{\boldsymbol{A}}(b_0, \ldots, b_{n-1})) \in F;$$

(v) $a, \mathrm{eq}(a, b) \in F$ implies $b \in F$ (*detachment*);

(vi) $a, b \in F$ implies $\mathrm{eq}(a, b) \in F$ (*G-rule*).

The next theorem gives a useful characterization of $\mathscr{L}$-admissible operations in terms of the relation between $\mathscr{L}$-filters and congruences. Two binary operations eq

and eq′ on $\boldsymbol{A}$ are said to be *$\mathscr{L}$-equivalent* if

$$\mathrm{eq}(a, b) \in F \text{ iff } \mathrm{eq}'(a, b) \in F,$$

for every $\mathscr{L}$-filter and all $a, b \in A$. It is easy to see that in this case eq is $\mathscr{L}$-admissible iff eq′ is.

Let Co $\boldsymbol{A}$ denote the set of all congruence relations on $\mathbf{A}$. It forms an algebraic closed-set system (see the Introduction) that contains the identity relation $\Delta_A$ and the universal relation $\nabla_A$ as its smallest and largest members, respectively.

A mapping $\Phi$ between two closure systems $\mathscr{C}$ and $\mathscr{C}'$ is said to *preserve closed unions* if, for any family $C_i$, $i \in I$, of closed sets in $\mathscr{C}$ such that $\bigcup C_i \in \mathscr{C}$, we have $\Phi(\bigcup C_i) = \bigcup \Phi(C_i)$.

**Theorem 4.2.** *Let $\mathscr{L}$ be a standard logic and $\boldsymbol{A}$ a $\Lambda$-algebra. There is a one-one correspondence between $\mathscr{L}$-equivalence classes of $\mathscr{L}$-admissible operations on $\boldsymbol{A}$ and mappings $\Omega : \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A} \to \mathrm{Co}\ \boldsymbol{A}$ such that:*

(i) *$\Omega$ preserves intersections and closed unions;*

(ii) $[\mathrm{t}](\Omega(F)) = F$ *for every* $F \in \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A}$.

*Moreover, if* eq *is an $\mathscr{L}$-admissible operation on $\boldsymbol{A}$, the corresponding mapping $\Omega$ is defined by*

$$\Omega(F) := \{\langle a, b\rangle \in A \times A : \mathrm{eq}(a, b) \in F\}. \tag{4.1}$$

**Proof.** Let eq be an $\mathscr{L}$-admissible operation on $\boldsymbol{A}$, and let $\Omega$ be defined as in (4.1). $\Omega(F)$ is a congruence by Definition 4.1(i)–(iv). We show that $F = [\mathrm{t}](\Omega(F))$.

Suppose $a \in [\mathrm{t}](\Omega(F))$. Thus $\mathrm{t} \equiv a(\Omega(F))$, i.e., $\mathrm{eq}(\mathrm{t}, a) \in F$, and hence $a \in F$ by detachment (Definition 4.1(iv)); recall that t is contained in every filter since $\mathscr{L}$ is standard. Conversely, if $a \in F$, then $\mathrm{eq}(a, \mathrm{t}) \in F$ by the G-rule (Definition 4.1(v)), and hence $a \equiv \mathrm{t}(\Omega)$; so $a \in [\mathrm{t}](\Omega(F))$.

It follows immediately from its definition (4.1) that $\Omega$ preserves intersections and closed unions. We show this only for closed unions. Let $F_i$, with $i \in I$, be any system of $\mathscr{L}$-filters such that $\bigcup F_i$ is an $\mathscr{L}$-filter.

$$\begin{aligned} a \equiv b(\Omega(\textstyle\bigcup F_i)) &\text{ iff } \mathrm{eq}(a, b) \in \textstyle\bigcup F_i \\ &\text{iff } \mathrm{eq}(a, b) \in F_i, \quad \text{for some } i \\ &\text{iff } a \equiv b(\Omega(F_i)), \quad \text{for some } i \\ &\text{iff } a \equiv b(\textstyle\bigcup \Omega(F_i)). \end{aligned}$$

Now assume $\Omega : \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A} \to \mathrm{Co}\ \boldsymbol{A}$ satisfies conditions (i) and (ii). For all $a, b \in A$ let

$$F(a, b) := \bigcap \{G \in \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A} : a \equiv b(\Omega(G))\}.$$

$F(a, b)$ is an $\mathscr{L}$-filter since the $\mathscr{L}$-filters form a closed-set system. Since $\Omega$ preserves intersections,

$$\Omega(F(a, b)) = \bigcap \{\Omega(G) : G \in \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A},\ a \equiv b(\Omega(G))\}.$$

Hence $a \equiv b(\Omega(F(a, b)))$. We claim $F(a, b)$ is a principal $\mathscr{L}$-filter (i.e., that it is generated by a single element). Suppose not. Then

$$F(a, b) = \bigcup \{G \in \mathrm{Fi}_{\mathscr{L}} A: G \subset F(a, b)\}$$

("$\subset$" means strict inclusion). For if the union on the right were strictly included in $F(a, b)$, then any element of $F(a, b)$ that is not contained in the union would be a principal generator of $F(a, b)$. Since $\Omega$ preserves closed unions,

$$\Omega(F(a, b)) = \bigcup \{\Omega(G): G \in \mathrm{Fi}_{\mathscr{L}} A, G \subset F(a, b)\}.$$

Thus $a \equiv b(\Omega(F(a, b)))$ implies $a \equiv b(\Omega(G))$ for some $\mathscr{L}$-filter $G \subset F(a, b)$, which contradicts the definition of $F(a, b)$ as the smallest $\mathscr{L}$-filter $G$ such that $a \equiv b(\Omega(G))$. So $F(a, b)$ is principal.

Let eq be a choice function that chooses a principal generator $\mathrm{eq}(a, b)$ of $F(a, b)$ for all $a, b \in A$. (Clearly any two such choice functions are $\mathscr{L}$-equivalent.) Notice first of all that, for any $\mathscr{L}$-filter $G$,

$$\mathrm{eq}(a, b) \in G \text{ iff } F(a, b) \subseteq G \text{ iff } a \equiv b(\Omega(G)). \tag{4.2}$$

We verify conditions (i)–(vi) of the Definition 4.1 of $\mathscr{L}$-admissibility for every $\mathscr{L}$-filter $G$ of $\boldsymbol{A}$.

Definition 4.1(i)–(iv) are straightforward. Consider for example Definition 4.1(iii). Assume $\mathrm{eq}(a, b), \mathrm{eq}(b, c) \in G$. Then by (4.2), $a \equiv b(\Omega(G))$ and $b \equiv c(\Omega(G))$. So $a \equiv c(\Omega(G))$, and hence $\mathrm{eq}(a, c) \in G$, again by (4.2).

Consider Definition 4.1(vi) (G-rule). Suppose $a, b \in G$. Since $G = [\mathrm{t}](\Omega(G))$, $a \equiv \mathrm{t} \equiv b(\Omega(G))$. So $\mathrm{eq}(a, b) \in G$.

Consider Definition 4.1(v) (detachment). Suppose $a$, $\mathrm{eq}(a, b) \in G$. Then by the G-rule, $\mathrm{eq}(\mathrm{t}, a) \in G$ ($\mathrm{t} \in G$ since $\mathscr{L}$ is standard). So, by Definition 4.1(iii), $\mathrm{eq}(\mathrm{t}, b) \in G$. Thus $b \equiv \mathrm{t}(\Omega(G))$, which means $b \in [\mathrm{t}](\Omega(G)) = G$. □

This theorem has a number of interesting consequences. For each $\mathscr{L}$-filter $F$ the mapping $\Omega$ chooses one of the possibly many congruences $\Theta$ with the property that $[t](\Theta) = F$. The particular choice it makes depends on the particular $\mathscr{L}$-admissible operation eq. We see in the following corollary that in at least one important case the mapping $\Omega$ is canonical.

An operation eq is said to be *compatible* with $\Theta \in \mathrm{Co}\, \boldsymbol{A}$ if $a \equiv a'(\Theta)$ and $b \equiv b'(\Theta)$ imply $\mathrm{eq}(a, b) \equiv \mathrm{eq}(a', b')(\Theta)$. Observe that a term-definable operation is automatically compatible with every congruence of $\boldsymbol{A}$.

**Corollary 4.3.** *Assume* eq *is a term-definable $\mathscr{L}$-admissible operation on $\boldsymbol{A}$ (or, more generally, an $\mathscr{L}$-admissible operation that is compatible with every congruence of $\boldsymbol{A}$). Let $F \in \mathrm{Fi}_{\mathscr{L}} A$. Then $\Omega(F)$ can be characterized as the largest congruence $\Theta$ such that $[\mathrm{t}](\Theta) = F$. All term-definable $\mathscr{L}$-admissible operations on $\boldsymbol{A}$ are $\mathscr{L}$-equivalent.*

**Proof.** Let $\Theta$ be any congruence such that $[\mathrm{t}](\Theta) = F$, and suppose $a \equiv b(\Theta)$. Then $\mathrm{eq}(a, b) \equiv_\Theta \mathrm{eq}(a, a) \equiv_\Theta \mathrm{t}$. So $\mathrm{eq}(a, b) \in F$, and hence $a \equiv b(\Omega(F))$.

The second part of the corollary follows immediately from the theorem. □

**Definition 4.4.** Let $\mathscr{L}$ be a standard logic and $\boldsymbol{A}$ a $\Lambda$-algebra. A congruence $\Theta$ on $\boldsymbol{A}$ is called an *$\mathscr{L}$-congruence* if

(i) $[t](\Theta)$ is an $\mathscr{L}$-filter;

(ii) $\Theta$ is the largest $\Phi \in \mathrm{Co}\,\boldsymbol{A}$ such that $[t](\Phi) = [t](\Theta)$.

The set of all $\Lambda$-congruences is denoted by $\mathrm{Co}_{\mathscr{L}}\boldsymbol{A}$.

If $\boldsymbol{A}$ has a term-definable $\mathscr{L}$-admissible operation, then it follows from Corollary 4.3 that $\mathrm{Co}_{\mathscr{L}}\boldsymbol{A}$ is a closed-set system, and, like every closed-set system, it forms a complete lattice whose meet operation is set-theoretical intersection and whose join is given by

$$\bigvee \Theta_i = \bigcap \{\Phi \in \mathrm{Co}_{\mathscr{L}}\boldsymbol{A} \colon \Theta_i \subseteq \Phi \text{ for all } i\}.$$

**Corollary 4.5.** *Assume $\boldsymbol{A}$ has a term-definable $\mathscr{L}$-admissible operation (or, more generally, an $\mathscr{L}$-admissible operation that is compatible with every congruence of $\boldsymbol{A}$). The mapping $\Omega : \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A} \to \mathrm{Co}_{\mathscr{L}}\boldsymbol{A}$ defined in* (4.1) *is an isomorphism between the lattices of $\mathscr{L}$-filters and $\mathscr{L}$-congruences that preserves closed unions. Its inverse is $\Theta \mapsto [t](\Theta)$.*

Of particular interest are those logics $\mathscr{L}$ for which there is an $\mathscr{L}$-admissible operation uniformly term-definable over all $\Lambda$-algebras.

**Definition 4.6.** A standard logic $\mathscr{L}$ is said to be *autoalgebraizable* if there exists a term $\mathsf{eq} \in \mathrm{Te}_{\Lambda}(x, y)$ such that $\mathsf{eq}^{\boldsymbol{A}}$ is an $\mathscr{L}$-admissible operation on $\boldsymbol{A}$ for every $\Lambda$-algebra $\boldsymbol{A}$. eq is called an *admissible term* for $\mathscr{L}$.

The notion of an autoalgebraizable logic is a special case of that of an *algebraizable* logic studied in [4].

From Corollary 4.3 we have that, if $\mathscr{L}$ is autoalgebraizable, any two $\mathscr{L}$-admissible terms are $\mathscr{L}$-equivalent in the obvious sense. Furthermore, by Corollary 4.5, there is a natural one-one correspondence (given by the mapping $\Omega$ defined in (4.1)) between the $\mathscr{L}$-filters and $\mathscr{L}$-congruences of each algebra of signature $\Lambda$.

We now want to prove an algebraic representation theorem that will allow us to obtain in the next section an analogue of Theorem 1.3 for autoalgebraizable logics. We first make a useful definition.

**Definition 4.7.** Let $\mathscr{L}$ be an autoalgebraizable logic, and let eq be an $\mathscr{L}$-admissible term for $\mathscr{L}$. An algebra of signature $\Lambda$ is called an *$\mathscr{L}$-algebra* if

(i) $\{t\}$ is an $\mathscr{L}$-filter of $\boldsymbol{A}$, and

(ii) $\Omega(\{t\})$ is the identity relation $\Delta_A$, i.e., $\Delta_A$ is the only congruence $\Theta$ such that $[t](\Theta) = \{t\}$.

Note that $\boldsymbol{A}$ is an $\mathscr{L}$-algebra iff $\Delta_A$ is an $\mathscr{L}$-congruence. It is also easy to verify that a congruence $\Theta$ on an arbitrary algebra $\boldsymbol{A}$ of signature $\Lambda$ is an $\mathscr{L}$-congruence iff the quotient $\boldsymbol{A}/\Theta$ is an $\mathscr{L}$-algebra.

$\mathscr{CPL}$-algebras and Boolean algebras are the same. We shall see that, for an arbitrary autoalgebraizable logic with the deduction theorem, $\mathscr{L}$-algebras play the same role as Boolean algebras do for $\mathscr{CPL}$. The $\mathscr{CPL}$-filters on a Boolean algebra are the Boolean filters in the usual sense. In this case the mapping

$$F \mapsto \Omega(F) := \{\langle a, b\rangle : (\text{not } a \text{ or } b) \text{ and } (a \text{ or not } b)\}$$

establishes a one-one correspondence between Boolean filters and all congruence relations.

The following characterization of $\mathscr{L}$-algebras in terms of conditional equations is straightforward.

**Proposition 4.8.** *Let $\mathscr{L} = \langle \Lambda, \vdash_r \rangle$ be an autoalgebraizable logic with $\mathscr{L}$-admissible term* eq. *An algebra of signature $\Lambda$ is an $\mathscr{L}$-algebra iff it satisfies the following conditional equations.*

(i) $t \approx \mathrm{t}$ *for every axiom* $t$.

(ii) $t_0 \approx \mathrm{t} \wedge \cdots \wedge t_{n-1} \approx \mathrm{t} \to r \approx \mathrm{t}$ *for every rule* $\frac{t_0, \ldots, t_{n-1}}{r}$,

(iii) $\mathrm{eq}(x, y) \approx \mathrm{t} \to x \approx y$.

**Proof.** Conditions (i) and (ii) together say that $\{\mathrm{t}\}$ is an $\mathscr{L}$-filter. Condition (iii) says that $\Omega(\{\mathrm{t}\}) = \Delta_A$. □

The next theorem is the main result of the section. An algebra $\boldsymbol{A}$ is a *subdirect product* of a system $\{\boldsymbol{B}_i : i \in I\}$ of algebras if $\boldsymbol{A}$ is a subalgebra of the direct product $\prod \boldsymbol{B}_i$ such that the projection of $\boldsymbol{A}$ on each factor $\boldsymbol{B}_i$ is surjective.

**Theorem 4.9.** *Assume $\mathscr{L}$ is a standard logic and is defined by the set* **L** *of standard matrices. Assume further that $\mathscr{L}$ is autoalgebraizable and that every protomatrix in* **L** *is an $\mathscr{L}$-algebra. Then every $\mathscr{L}$-algebra* **A** *is isomorphic to a direct limit of subdirect products of protomatrices in* **L**. *If* **L** *is a finite set of finite matrices, then* **A** *itself is isomorphic to a subdirect product of protomatrices in* **L**.

**Proof.** Because of the isomorphism $\Omega$ between the lattices of $\mathscr{L}$-filters and $\mathscr{L}$-congruences, in dealing with quotients in the class of algebras of signature $\Lambda$ we can replace $\mathscr{L}$-congruences by $\mathscr{L}$-filters. In particular, for any algebra $\boldsymbol{A}$ and $\mathscr{L}$-filter $F$ on $\boldsymbol{A}$, $\boldsymbol{A}/F$ will mean the quotient algebra $\boldsymbol{A}/\Omega(F)$. Note that, if $\boldsymbol{A}$ is an $\mathscr{L}$-algebra, we have $\boldsymbol{A}/\{\mathrm{t}\} = \boldsymbol{A}/\Omega(\{\mathrm{t}\}) = \boldsymbol{A}/\Delta_A \cong \boldsymbol{A}$. Moreover, the homomorphism theorem of universal algebra [16, p. 57] holds in the usual way when congruences are replaced by filters, provided the codomain of the homomorphism is an $\mathscr{L}$-algebra: more precisely, let $h : \boldsymbol{A} \to \boldsymbol{B}$ be a surjective homomorphism from an arbitrary $\Lambda$-algebra $\boldsymbol{A}$ onto an $\mathscr{L}$-algebra $\boldsymbol{B}$, and let $F = h^{-1}(\{\mathrm{t}\}) := \{a \in A : h(a) = \mathrm{t}\}$. Then $F$ is an $\mathscr{L}$-filter since $\{\mathrm{t}\}$ is, and $\boldsymbol{A}/F \cong \boldsymbol{B}/\{\mathrm{t}\} \cong \boldsymbol{B}$.

An $\mathscr{L}$-filter $G$ of $\mathrm{Te}_\Lambda(X)$ is *primitive if* $\mathrm{Te}_\Lambda(X)/G \cong \boldsymbol{L}$ for some $\boldsymbol{L} \in \mathbf{L}$. If $F = \bigcap G_i$, where the $G_i$ are primitive filters, then $\mathrm{Te}_\Lambda(X)/F$ is isomorphic to a subdirect product of protomatrices in **L** [16, p. 123].

Let $\boldsymbol{A}$ be an arbitrary $\mathscr{L}$-algebra. To simplify the argument we assume $\boldsymbol{A}$ is countable. Let $h:\mathbf{Te}_{\Lambda}(X)\to\boldsymbol{A}$ be a surjective homomorphism and let $H$ be its kernel. Note $\mathbf{Te}_{\Lambda}(X)/H\cong\boldsymbol{A}$. Let $t_0,\ldots,t_{n-1}\in H$ and $r\in\mathbf{Te}_{\Lambda}(X)\setminus H$. Then $r\notin[t_0,\ldots,t_{n-1}]_{\mathscr{L}}$ since $H$ is an $\mathscr{L}$-theory, and hence $t_0,\ldots,t_{n-1}\not\vdash_{\mathscr{L}}\mathrm{r}$ (see (2.2)). So there exists an $\boldsymbol{L}\in\mathbf{L}$ and $g:\mathbf{Te}_{\Lambda}(X)\to\boldsymbol{L}$ such that $g(t_1)=\cdots=g(t_{n-1})=\mathrm{t}$, but $g(r)\notin\mathrm{t}$. Thus the kernel $G$ of $g$ is a primitive $\mathscr{L}$-filter such that $t_0,\ldots,t_{n-1}\in G$ but $r\notin G$. For any finite subset $T$ of terms in $\mathbf{Te}_{\Lambda}(X)$, let $F(T)$ be the intersection of all primitive filters that include $T$. If $T\subseteq H$, then by the above, $T\subseteq F(T)\subseteq H$ for all $T$. Let $H^*$ be the set of all finite subsets of $H$. Then $H=\bigcup_{T\in H^*}F(T)$. $H^*$ forms a directed partially ordered set under inclusion, and for each pair $T$, $T'$ of finite subsets of $H$ such that $T\subseteq T'$, we have $F(T)\subseteq F(T')$. Let

$$f_{T,T'}:\mathbf{Te}_{\Lambda}(X)/F(T)\to\mathbf{Te}_{\Lambda}(X)/F(T')$$

be the natural surjection. Let $\boldsymbol{B}_T=\mathbf{Te}_{\Lambda}(X)/F(T)$. Note that $\boldsymbol{B}_T$ is isomorphic to a subalgebra of a direct product of protomatrices of $\mathbf{L}$. Clearly, the set of $\boldsymbol{B}_T$ with $T\in H^*$, together with the homomorphisms $f_{T,T'}$ with $T\subseteq T'$, forms a directed family of $\Lambda$-algebras. Let $\boldsymbol{B}=(\bigcup\boldsymbol{B}_T)/\equiv$ (where the union is disjoint) be the direct limit of this family. (See for instance [16, p. 129].) It is a routine matter to show that $t\mapsto[[t](F(\mathrm{t}))](\equiv)$ is a surjective homomorphism from $\mathbf{Te}_{\Lambda}(X)$ onto $\boldsymbol{B}$ with kernel $H$. Thus $\boldsymbol{B}\cong\boldsymbol{A}$. This proves the first part of the theorem.

Assume now that $\mathbf{L}$ is a finite set of finite matrices. Let $\boldsymbol{A}$ and $h:\mathbf{Te}_{\Lambda}(X)\to\boldsymbol{A}$ be as above, and again let $H$ be the kernel of $h$. Let $t_1, t_2, \ldots, t_n, \ldots$ be an enumeration of the terms in $H$, and set $H_n:=\{t_1, t_2, \ldots, t_n\}$ for each $n<\omega$. Let $X_n$ be the set of variables occurring in $t_1,\ldots,t_n, r$. A mapping $g:\mathbf{Te}_{\Lambda}(X_n)\to\boldsymbol{L}$ is called *good* if $\boldsymbol{L}\in\mathbf{L}$ and $g(t_1)=\cdots=g(t_n)=\mathrm{t}$ while $g(r)\neq\mathrm{t}$. Because of the assumption about $\mathbf{L}$, there are only finitely many good mappings for each $n$. Thus at least one of them can be extended to a good map for each $m\geq n$. So there exits a $g:\mathbf{Te}_{\Lambda}(X)\to\boldsymbol{L}$ for some $\boldsymbol{L}\in\mathbf{L}$ that is good for all $n$. Its kernel is a primitive $\mathscr{L}$-filter that includes $H$ but does not contain $r$. Choosing one such primitive filter for each $r$ not in $H$, we obtain a representation of $\boldsymbol{A}$ as a subdirect product of protomatrices in $\mathbf{L}$. □

### *4.1. Equality-test operations*

There is a simple property of the set of defining matrices of a standard logic that if verified guarantees that the logic is autoalgebraizable. It involves the notion of an equality-test which we now finally define.

**Definition 4.10.** Let $\boldsymbol{A}$ be a $\Lambda$-algebra. An operation $\mathrm{eq}:A\times A\to A$ is called an *equality-test operation* on $\boldsymbol{A}$ if, for all $a, b\in A$, we have $\mathrm{eq}(a, b)=\mathrm{t}$ iff $a=b$.

Note that the conditional equation of Proposition 4.8(iii) is sufficient for an $\mathscr{L}$-admissible operation eq to be an equality-test; it is always a necessary condition. Thus for autoalgebraizable logics with $\mathscr{L}$-admissible term eq, an algebra $\boldsymbol{A}$ is an $\mathscr{L}$-algebra iff $\{\mathrm{t}\}$ is an $\mathscr{L}$-filter and $\mathrm{eq}^{\boldsymbol{A}}$ is an equality-test operation on $\boldsymbol{A}$.

**Theorem 4.11.** *Let $\mathscr{L}$ be a standard logic and let* **L** *be the set of standard matrices that defines it. Let* $\mathbf{eq}(x, y) \in \mathrm{Te}_{\Lambda}(x, y)$. *If* $\mathbf{eq}^{L}$ *is an equality-test operation for every protomatrix* $L \in \mathbf{L}$, *then $\mathscr{L}$ is autoalgebraizable and* **eq** *is an $\mathscr{L}$-admissible term for $\mathscr{L}$.*

**Proof.** Let $A$ be any $\Lambda$-algebra. We verify the defining conditions of Definition 4.1(i)–(vi) of $\mathscr{L}$-admissibility. Notice that each of these conditions corresponds in the obvious way to an $\mathscr{L}$-entailment relation to which it is equivalent. Consider for example Definition 4.1(ii):

$$\mathbf{eq}^{A}(a, b), \mathbf{eq}^{A}(b, c) \in F \text{ implies } \mathbf{eq}^{A}(a, c) \in F.$$

The condition that this holds for every $\mathscr{L}$-algebra $A$ and every $\mathscr{L}$-filter of $A$ is equivalent to the entailment

$$\mathbf{eq}(x, y), \mathbf{eq}(y, z) \vdash_{\mathscr{L}} \mathbf{eq}(x, z). \tag{4.3}$$

This depends of course on the assumption $\mathbf{eq} \in \mathrm{Te}_{\Lambda}(x, y)$. Since $\mathscr{L}$ is standard, (4.3) is in turn equivalent to the condition that the conditional equation

$$\mathbf{eq}(x, y) \approx \mathbf{t} \wedge \mathbf{eq}(y, z) \approx \mathbf{t} \rightarrow \mathbf{eq}(x, z) \approx \mathbf{t}$$

holds identically in each of the protomatrices $L$ defining $\mathscr{L}$. But this is obvious since by assumption $\mathbf{eq}^{L}$ is an equality-test operation for $L$. All the defining conditions of $\mathscr{L}$-admissibility can be verified this way. □

This theorem has a weak form of converse:

**Proposition 4.12.** *Let $\mathscr{L}$ be an autoalgebraizable standard logic with $\mathscr{L}$-admissible term* **eq**. *Then $\mathscr{L}$ can be defined by a set* **L** *of standard matrices such that* $\mathbf{eq}^{L}$ *is an equality-test operation on each protomatrix $L$ of* **L**; *i.e., $L$ is an $\mathscr{L}$-algebra.*

**Proof.** Let **M** be any defining set of standard matrices for $\mathscr{L}$. Let $M$ be a protomatrix of **M** and set $\Theta_M := \Omega(\{\mathbf{t}\})$. Finally, let **L** be the corresponding set of quotient matrices $\langle M/\Theta_M, \{[\mathbf{t}](\Theta_M)\}\rangle$. By Corollary 4.3, $\Theta_M$ is the largest congruence on $M$ with the property that its t-equivalence class contains only t, i.e., $[\mathbf{t}](\Theta_M) = \{\mathbf{t}\}$.

Let $t(\bar{x})$ be any term in $\mathrm{Te}_{\Lambda}(X)$ and $\bar{a}$ any assignment in $M$. Let $h: M \rightarrow M/\Theta_M$ be the natural surjective homomorphism, and take $h(\bar{a})$ to be the assignment in $M/\Theta_M$ that assigns $[a](\Theta_M)$ to a variable $x$ if $\bar{a}$ assigns $a$ to $x$. Since t is $\Theta_M$-equivalent only to itself, it is easy to see that $t^{M/\Theta_M}(h(\bar{a})) = \mathbf{t}$ iff $t^{M}(\bar{a}) = \mathbf{t}$. Thus $t_0, \ldots, t_{n-1} \models_{\mathbf{M}} r$ iff $t_0, \ldots, t_{n-1} \models_{\mathbf{L}} r$, for all $t_0, \ldots, t_{n-1}, r \in \mathrm{Te}_{\Lambda}(X)$. This shows **L** also defines $\mathscr{L}$. Each protomatrix in **L** is an $\mathscr{L}$-algebra. □

The paradigm for autoalgebraizable logics is $\mathscr{CPL}$.

$$\mathbf{eq}(x, y) := (\text{not } x \text{ or } y) \text{ and } (x \text{ or not } y) \tag{4.4}$$

is a term-definable equality-test operation for $\mathscr{CPL}$. This follows immediately from the last theorem since **eq** is an equality-test term for $\boldsymbol{B}_2$.

It is easy to see that

$$\mathsf{eq}(x, y) := (x \to y) \text{ and } (y \to x) \tag{4.5}$$

is an equality-test term for $\mathbf{LU}_3$. Thus Łukasiewicz's 3-valued logic is autoalgebraizable. More generally, any standard logic that is defined by a single primal protomatrix is autoalgebraizable.

The term (4.5) also defines an equality-test operation on the protomatrix $\mathbf{HE}_{\omega+1}$. Thus $\mathscr{L}\mathscr{J}\mathscr{L}$ is autoalgebraizable. On the other hand, it is easy to check that none of the protomatrices $\mathbf{KW}_3$, $\mathbf{KS}_3$, $\mathbf{MC}_3$, or $\mathbf{BE}_4$ has a term-definable equality-test.

### *4.2. The deduction theorem in autoalgebraizable logics*

We will show that if the deduction theorem holds in an autoalgebraizable logic, then the property of $\mathscr{L}$-theories expressed in Proposition 2.1 applies to the $\mathscr{L}$-filters of every algebra. Recall that, for any $\mathscr{L}$-filter $F$, $F[a]$ is the $\mathscr{L}$-filter generated by $F \cup \{a\}$.

**Proposition 4.13.** *Let $\mathscr{L}$ be a autoalgebraizable standard logic, and assume that the deduction theorem holds for $\mathscr{L}$ with deduction term* de. *Let $\mathbf{A}$ be a $\Lambda$-algebra and $a, b \in A$. Then for any $\mathscr{L}$-filter $F$ of $\mathbf{A}$,*

$$b \in F[a] \text{ iff } \mathsf{de}^{\mathbf{A}}(a, b) \in F.$$

To prove this proposition we need the following.

**Lemma 4.14.** *Let $\mathscr{L}$ be a autoalgebraizable standard logic. Let $\mathbf{A}$ be a $\Lambda$-algebra and $h: \mathbf{Te}_\Lambda(X) \to \mathbf{A}$ a surjective homomorphism. Let $F$ be an $\mathscr{L}$-filter of $\mathbf{A}$ and set $T := h^{-1}(F)$, so that $T$ is an $\mathscr{L}$-theory. For any $a \in A$ let $t \in \mathrm{Te}_\Lambda(X)$ such that $h(t) = a$. Then $h^{-1}(F[a]) = T[t]$.*

**Proof.** This is a consequence of the composition of two correspondences: the one between $\mathscr{L}$-filters and $\mathscr{L}$-congruences and the one between the congruences on an arbitrary algebra and the congruences on any one of its quotients.

According to the correspondence theorem of universal algebra (see for instance [16, p. 61, Theorem 3]), the mappings

$$\Theta \mapsto h^{-1}(\Theta) := \{\langle t, r\rangle \in \mathrm{Te}_\Lambda(X)^2 : h(t) \equiv h(r)(\Theta)\},$$

$$\Phi \mapsto h(\Phi) := \{\langle h(t), h(r)\rangle : t \equiv r(\Phi)\}$$

are mutually inverse order-preserving, one-one correspondences between the set of all congruences of $\mathbf{A}$ and the set of congruences of $\mathbf{Te}_\Lambda(X)$ that include $h^{-1}(\Delta_A)$ (the *congruence-kernel* of $h$). Moreover, $\mathbf{A}/\Theta \cong \mathbf{Te}_\Lambda(X)/h^{-1}(\Theta)$. This isomorphism means that $\Theta$ is an $\mathscr{L}$-congruence iff $h^{-1}(\Theta)$ is one. Because of the correspondence between $\mathscr{L}$-filters and $\mathscr{L}$-congruences, the mappings $G \mapsto h^{-1}(G)$ and $T \mapsto h(T)$ are mutually inverse order-preserving, one-one correspondences between $\mathscr{L}$-filters and $\mathscr{L}$-theories that include $h^{-1}(\mathrm{t})$.

We now prove the lemma. $h^{-1}(F[a])$ is a theory that includes $T \cup \{t\}$, so $T[t] \subseteq h^{-1}(F[a])$. Consider any term $r$ such that $r \notin T[t]$. Let $T'$ be a theory such that $T \cup \{t\} \subseteq T'$ but $r \notin T'$. By the above correspondence there is an $\mathscr{L}$-filter $F'$ of $\boldsymbol{A}$ such that $h^{-1}(F') = T'$ and $h(T') = F'$. Note that $F \cup \{a\} = h(T \cup \{t\}) \subseteq h(T') = F'$. Thus $F[a] \subseteq F'$, and hence $h^{-1}(F[a]) \subseteq h^{-1}(F') = T'$. So $r \notin h^{-1}(F[a])$. Since this holds for all $r \notin T[t]$, we have $h^{-1}(F[a]) \subseteq T[t]$. □

The proof of the proposition is a straightforward consequence of the lemma and Proposition 2.1. □

By iterating the application of the deduction term de we can obtain a useful extension of the proposition. Let $\mathsf{de}_1(x_0, y) := \mathsf{de}(x_0, y)$ and

$$\mathsf{de}_{n+1}(x_0, \ldots, x_n, y) := \mathsf{de}(x_0, \mathsf{de}_n(x_0, \ldots, x_{n-1}, y))$$

Then under the hypothesis of the proposition,

$$b \in F[a_0, \ldots, a_{n-1}] \text{ iff } \mathsf{de}^{\boldsymbol{A}}_{n-1}(a_0, \ldots, a_{n-1}, b) \in F;$$

in particular,

$$b \in [a_0, \ldots, a_{n-1}] \text{ iff } \mathsf{de}^{\boldsymbol{A}}_{n-1}(a_0, \ldots, a_{n-1}, b) = \mathsf{t}.$$

We remark that the assumption in Proposition 4.13 that $\mathscr{L}$ be standard can be considerably weakened.

## 5. Generalized equality-test algebras over an autoalgebraizable logic

Throughout this section $\mathscr{L}$ will be a standard logic for which we have an axiomatization (a set of axioms and rules of inference). We assume $\mathscr{L}$ is autoalgebraizable and that $\mathsf{eq}_{\log}$ is an admissible term for $\mathscr{L}$. We also assume that, if $\mathbf{L}$ is a set of defining standard matrices for $\mathscr{L}$, then $\mathsf{eq}^{\boldsymbol{L}}_{\log}$ is an equality-test operation for every protomatrix $\boldsymbol{L} \in \mathbf{L}$, i.e., each $\boldsymbol{L}$ is an $\mathscr{L}$-algebra; by Proposition 4.12 this results in no loss of generality. Finally, we assume that the deduction theorem holds for $\mathscr{L}$ with deduction term de.

We now give the axioms of generalized admissible equality-test algebras over $\mathscr{L}$; they should be compared with the AXGET axioms in Section 1.

($\mathrm{Axget}_{\mathscr{L},1}$) $t \approx \mathsf{t}$ for every axiom $t$;

($\mathrm{Axget}_{\mathscr{L},1'}$) $t_0 \approx \mathsf{t} \wedge \cdots \wedge t_{n-1} \approx \mathsf{t} \to r \approx \mathsf{t}$ for each inference rule $\frac{t_0, \ldots, t_{n-1}}{r}$;

($\mathrm{Axget}_{\mathscr{L},1''}$) $\mathsf{eq}_{\log}(x, y) \approx \mathsf{t} \to x \approx y$.

For each $s \in S$:

($\mathrm{Axget}_{\mathscr{L},2}$) $\mathsf{eq}_s(x, x) \approx \mathsf{t}$;

($\mathrm{Axget}_{\mathscr{L},3}$) $\mathsf{de}(\mathsf{eq}_s(x, y), \mathsf{eq}_s(y, x)) \approx \mathsf{t}$;

($\mathrm{Axget}_{\mathscr{L},4}$) $\mathsf{de}_2(\mathsf{eq}_s(x, y), \mathsf{eq}_s(y, z), \mathsf{eq}_s(x, z)) \approx \mathsf{t}$.

For each $\sigma \in \Sigma_O$ of type $s_0 \dots s_{n-1} \to s$:

$(\text{Axget}_{\mathscr{L},5})$ $\mathsf{de}_n(\mathsf{eq}_{s_0}(x_0, y_0), \dots, \mathsf{eq}_{s_{n-1}}(x_{n-1}, y_{n-1})$

$$\mathsf{eq}_s(\sigma(x_0, \dots, x_{n-1}), \sigma(y_0, \dots, y_{n-1}))) \approx \mathsf{t}.$$

For each $\sigma \in \Sigma_P$ of type $s_0 \dots s_{n-1}$:

$(\text{Axget}_{\mathscr{L},6})$ $\mathsf{de}_{n+1}(\mathsf{eq}_{s_0}(x_0, y_0), \dots, \mathsf{eq}_{s_{n-1}}(x_{n-1}, y_{n-1}),$

$$\sigma(x_0, \dots, x_{n-1}), \sigma(y_0, \dots, y_{n-1})) \approx \mathsf{t}.$$

$(\text{Axget}_{\mathscr{L},7})$ $\mathsf{eq}_s(x, y) \approx \mathsf{t} \to x \approx y$;

$(\text{Axget}_{\mathscr{L},8})$ $\mathsf{de}_2(\mathsf{eq}_s(x_0, y_0), \mathsf{eq}_s(x_1, y_1), \mathsf{eq}_{\log}(\mathsf{eq}_s(x_0, x_1), \mathsf{eq}_s(y_0, y_1))) \approx \mathsf{t}.$

Let $\text{AXGET}_{\mathscr{L}}$ stand for the set of all these axioms.

**Definition 5.1.** A $\Sigma(\Lambda)$-algebra satisfying $\text{AXGET}_{\mathscr{L}}$ is called a *generalized admissible equality-test algebra over* $\mathscr{L}$ (a $\text{GET}_{\mathscr{L}}$ *algebra*).

The $\text{GET}_{\mathscr{L}}$ algebras form a conditional-equational class. Note that, apart from the axioms defining the underlying logic, $\text{Axget}_{\mathscr{L},7}$ is the only axiom that is not an equality.

**Proposition 5.2.** *A $\Sigma(\Lambda)$-algebra $\mathbf{A}$ is a $\text{GET}_{\mathscr{L}}$ algebra iff the following conditions hold:*

(i) $\mathbf{A}_{\log}$ *is an* $\mathscr{L}$*-algebra;*

(ii) $\{\mathsf{eq}_s^{\mathbf{A}} : s \in S\}$ *is an* $\mathscr{L}$*-admissible equality-test system for* $\mathbf{A}$;

(iii) *for every* $s \in S$ *and every* $\mathscr{L}$*-filter* $F$ *of* $\mathbf{A}_{\log}$,

$$\mathsf{eq}_s^{\mathbf{A}}(a_0, b_0), \mathsf{eq}_s^{\mathbf{A}}(a_1, b_1) \in F \text{ implies } \mathsf{eq}_{\log}^{\mathbf{A}}(\mathsf{eq}_s^{\mathbf{A}}(a_0, a_1), \mathsf{eq}_s^{\mathbf{A}}(b_0, b_1)) \in F.$$

**Proof.** $\text{Axget}_{\mathscr{L},1}$–$\text{Axget}_{\mathscr{L},1''}$ are the same three conditional equations that appear in Proposition 4.8. It is easy to see by Proposition 4.13 that $\text{Axget}_{\mathscr{L},2}$–$\text{Axget}_{\mathscr{L},7}$ are equivalent to (ii) and that $\text{Axget}_{\mathscr{L},8}$ is equivalent to (iii). □

Let $\mathbf{L}$ be the defining set of standard matrices for $\mathscr{L}$. By comparing this result with the definition of an admissible equality-test algebra over $\mathscr{L}$ given in Definition 3.3, we see that a special $\text{ET}_{\mathscr{L}}$ algebra is just a $\text{GET}_{\mathscr{L}}$ algebra whose $\Lambda$-reduct is one of the protomatrices of $\mathbf{L}$.

We are now finally ready for the nonclassical analogue of Theorem 1.3.

**Theorem 5.3.** *Let $\mathscr{L}$ be a standard logic defined by class $\mathbf{L}$ of standard matrices. Assume $\mathscr{L}$ is autoalgebraizable and that the deduction theorem holds for $\mathscr{L}$.*

(i) *Every $\text{GET}_{\mathscr{L}}$ algebra $\mathbf{A}$ is isomorphic to a direct limit of subdirect products of special $\text{ET}_{\mathscr{L}}$ algebras $\mathbf{B}$ such that $\mathbf{B}_{\log} \in \mathbf{L}$.*

(ii) *If* $\mathbf{L}$ *is a finite set of finite matrices, every* $\mathrm{GET}_{\mathscr{L}}$ *algebra* $\boldsymbol{A}$ *is isomorphic to a subdirect product of special* $\mathrm{ET}_{\mathscr{L}}$ *algebras* $\boldsymbol{B}$ *such that* $\boldsymbol{B}_{\log} \in \mathbf{L}$.

The proof will follow closely the proof of Theorem 4.9, the corresponding result for $\mathscr{L}$-algebras. The key to the proof is the notion of a $\mathrm{GET}_{\mathscr{L}}$-congruence. An $(S \cup \{\log\})$-sorted congruence $\Theta$ on a $\mathrm{GET}_{\mathscr{L}}$ algebra $\boldsymbol{A}$ is called a $\mathrm{GET}_{\mathscr{L}}$*-congruence* if the quotient $\boldsymbol{A}/\Theta$ is also a $\mathrm{GET}_{\mathscr{L}}$ algebra. All axioms in $\mathrm{AXGET}_{\mathscr{L}}$ except $\mathrm{Axget}_{\mathscr{L},1'}$, $\mathrm{Axget}_{\mathscr{L},1''}$, and $\mathrm{Axget}_{\mathscr{L},7}$ are equations and hence will automatically hold in every quotient of $\boldsymbol{A}$. Thus in order to show $\Theta$ is a $\mathrm{GET}_{\mathscr{L}}$-congruence it suffices to show that $(\boldsymbol{A}/\Theta)_{\log}$ is an $\mathscr{L}$-algebra and that each of the conditional equations $\mathrm{eq}_s(x, y) \approx \mathrm{t} \to x \approx y$ holds in $\boldsymbol{A}/\Theta$.

**Proof of Theorem 5.3.** Let $\mathrm{eq}_{\log}$ and de be an equality-test and deduction term for $\mathscr{L}$. In view of Proposition 4.12 we may assume without loss of generality that $\mathrm{eq}_{\log}^{\boldsymbol{L}}$ is an equality-test operation on each protomatrix $\boldsymbol{L} \in \mathbf{L}$.

Let $\boldsymbol{A}$ be an arbitrary $\mathrm{GET}_{\mathscr{L}}$ algebra. We modify the definition of the operator $\Omega$ defined in (4.1) of Theorem 4.2 in the obvious way and show that it is an isomorphism between the lattice of $\mathscr{L}$-filters of $\boldsymbol{A}_{\log}$ and the lattice of $\mathrm{GET}_{\mathscr{L}}$-congruences of $\boldsymbol{A}$. For each $F \in \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A}_{\log}$ and $s \in S \cup \{\log\}$ set

$$\Omega(F)_s := \{\langle a, b\rangle \in A_s^2 : \mathrm{eq}_s^{\boldsymbol{A}}(a, b) \in F\}$$

and $\Omega(F) := \langle \Omega(F)_s : s \in S \cup \{\log\}\rangle$. It follows from the $\mathscr{L}$-admissibility of $\{\mathrm{eq}_s^{\boldsymbol{A}} : s \in S\}$ and $\mathrm{eq}_{\log}^{\boldsymbol{A}}$, and from condition (iii) of Proposition 5.2, that $\Omega(F)$ is a congruence on $\boldsymbol{A}$.

Let $F \in \mathrm{Fi}_{\mathscr{L}}\boldsymbol{A}_{\log}$. $\Omega(F)_{\log}$ coincides with the congruence $\Omega(F)$ on $\boldsymbol{A}_{\log}$ defined in (4.1). Thus $(\boldsymbol{A}_A/\Omega(F))_{\log} := \boldsymbol{A}_{\log}/\Omega(F)_{\log}$ is an $\mathscr{L}$-algebra. To show $\Omega(F)$ is an $\mathrm{GET}_{\mathscr{L}}$-congruence it only remains to check that $\mathrm{eq}_s(x, y) \approx \mathrm{t} \to x \approx y$ holds in $\boldsymbol{A}/\Omega(F)$ for each $s \in S$. Consider any $a, b \in (\boldsymbol{A}/\Omega(F))_s$ and let $c, d \in A_s$ such that $a = [c](\Omega(F))$ and $b = [d](\Omega(F))$. Then $\mathrm{eq}_s^{\boldsymbol{A}/\Omega(F)}(a, b) = \mathrm{t}$ iff $\mathrm{eq}_s^{\boldsymbol{A}}(c, d) \in F$, which gives $c \equiv d(\Omega(F))$, i.e., $a = b$.

Conversely, let $\Theta$ be a $\mathrm{GET}_{\mathscr{L}}$-congruence, and let $F := [\mathrm{t}](\Theta_{\log})$. Since $\boldsymbol{A}_{\log}/\Theta_{\log}$ is an $\mathscr{L}$-algebra, $\Theta_{\log}$ is an $\mathscr{L}$-congruence, and hence $F$ is an $\mathscr{L}$-filter. Consider any $s \in S$ and $a, b \in A_s$. $a \equiv b(\Theta_s)$ implies

$$\mathrm{eq}_s^{\boldsymbol{A}}(a, b) \equiv_{\Theta_s} \mathrm{eq}_s^{\boldsymbol{A}}(a, a) = \mathrm{t},$$

i.e., $\mathrm{eq}_s^{\boldsymbol{A}}(a, b) \in F$. Conversely, $\mathrm{eq}_s^{\boldsymbol{A}}(a, b) \in F$ implies $\mathrm{eq}_s^{\boldsymbol{A}}(a, b) \equiv t(\Theta_s)$, which in turn implies $a \equiv b(\Theta_s)$ by $\mathrm{Axget}_{\mathscr{L},7}$. So $\Theta = \Omega(F)$.

Thus $\Omega$ is an isomorphism between the lattice of $\mathscr{L}$-filters of $A_{\log}$ and $\mathrm{GET}_{\mathscr{L}}$-congruences of $\boldsymbol{A}$. Composing the inverse of this isomorphism with the isomorphism in Corollary 4.5 we get an isomorphism between the lattice of $\mathrm{GET}_{\mathscr{L}}$-congruences of $\boldsymbol{A}$ and the lattice of $\mathscr{L}$-congruences of $\boldsymbol{A}_{\log}$. So each isomorphic representation of $\boldsymbol{A}$ as a subdirect product of $\mathrm{GET}_{\mathscr{L}}$ algebras corresponds in a natural way to an isomorphic representation of $\boldsymbol{A}_{\log}$ as a subdirect product of $\mathscr{L}$-algebras. Since the

special $\mathrm{ET}_{\mathscr{L}}$ algebras are exactly the $\mathrm{GET}_{\mathscr{L}}$ algebras whose $\Lambda$-reduct is a protomatrix of $\mathbf{L}$, the conclusion of the theorem follows immediately from Theorem 4.9. □

**Corollary 5.4.** *The* $\mathrm{GET}_{\mathscr{L}}$ *algebras form the smallest conditional-equational class that contains all special* $\mathrm{ET}_{\mathscr{L}}$ *algebras.*

The following analogue of Theorem 1.5 is an immediate consequence of Theorem 5.3 together with the equivalence (3.1). For any set $\Gamma$ of universal sentences let $\Gamma^* := \{\varphi^* \approx \mathrm{t}\colon \varphi \in \Gamma\}$. Recall that an admissible relational $\Sigma$-structure $\boldsymbol{R}$ over $\mathscr{L}$ is said to be *special* if its corresponding algebra-transform $\mathbf{Alg}\,\boldsymbol{R}$ is a special $\mathrm{ET}_{\mathscr{L}}$ algebra.

**Theorem 5.5.** *Let* $\mathscr{L}$ *be as in Theorem* 5.3. *Let* $\Gamma$ *be a set of universal* $\Sigma$*-sentences. Then* $\Gamma^* \cup \mathrm{AXGET}_{\mathscr{L}}$ *is a set of conditional-equational axioms for the smallest conditional-equational class that contains the algebra-transform* $\mathbf{Alg}\,\boldsymbol{R}$ *of every special admissible* $\Sigma$*-structure* $\boldsymbol{R}$ *over* $\mathscr{L}$ *such that* $\boldsymbol{R} \models_{\mathscr{L}} \Gamma$.

A $\mathrm{GET}_{\mathscr{L}}$ algebra $\boldsymbol{A}$ is *$\mathscr{L}$-simple* if $\{\mathrm{t}\}$ and $A_{\log}$ are the only $\mathscr{L}$-filters of $\boldsymbol{A}_{\log}$. This is a natural generalization of the classical GET-simple algebras discussed in Section 1 (see the remarks preceding Theorem 1.6). But the analogue of Theorem 1.6, that a $\mathrm{GET}_{\mathscr{L}}$ algebra is $\mathscr{L}$-simple iff it is a special $\mathrm{ET}_{\mathscr{L}}$ algebra, doesnot hold in general for a variety of different reasons, one of which is that a protomatrix of $\mathbf{L}$ need not be $\mathscr{L}$-simple. If we restrict ourselves to autoalgebraizable logics with a deduction term, and that are defined by a finite set of finite, $\mathscr{L}$-simple matrices, then the analogue of Theorem 1.6 holds. But very few logics satisfy this condition. For example among the nonclassical logics we have considered, only the 3-valued Łukasiewicz logic meets the criterion. $\mathscr{LJL}$ does not, and in fact the only extension of $\mathscr{LJL}$ that does is $\mathscr{CPL}$.

The notion of a *$\mathscr{L}$-subdirectly irreducible* algebra turns out to be the proper analogue of simplicity in the case of general nonclassical logics.

**Definition 5.6.** Let $\mathscr{L}$ be a autoalgebraizable standard logic and $\boldsymbol{A}$ a nontrivial $\mathscr{L}$-algebra. $\boldsymbol{A}$ is *$\mathscr{L}$-subdirectly irreducible* if $\mathrm{Fi}_{\mathscr{L}}\boldsymbol{A}\backslash\{\{\mathrm{t}\}\}$ contains a smallest member, i.e., $\boldsymbol{A}$ has a filter different from $\{\mathrm{t}\}$ that is included in every filter different from $\{\mathrm{t}\}$.

In an $\mathscr{L}$-subdirectly irreducible $\mathscr{L}$-algebra $\boldsymbol{A}$, the filter $\{\mathrm{t}\}$ cannot be represented as the intersection of a family of $\mathscr{L}$-filters unless one of them coincides with $\{\mathrm{t}\}$. Thus if $\boldsymbol{A}$ is isomorphic to a subdirect product of $\mathscr{L}$-algebras, it must be isomorphic to one of the factor algebras. This condition in fact characterizes $\mathscr{L}$-subdirectly irreducible $\mathscr{L}$-algebras.

A standard matrix is *$\mathscr{L}$-subdirectly irreducible* just when its protomatrix is; similarly a $\mathrm{GET}_{\mathscr{L}}$ algebra $\boldsymbol{A}$ is *$\mathscr{L}$-subdirectly irreducible* when $\boldsymbol{A}_{\log}$ is $\mathscr{L}$-subdirectly irreducible.

The next proposition is an analogue of a well known universal algebraic result of G. Birkhoff; see [16, p. 124].

**Proposition 5.7.** *Assume $\mathscr{L}$ is autoalgebraizable. Every $\mathscr{L}$-algebra is a subdirect product of $\mathscr{L}$-subdirectly irreducible $\mathscr{L}$-algebras.*

**Proof.** Consider any element $a \in A$ such that $a \neq \mathrm{t}$. The set of all $\mathscr{L}$-filters that fail to contain $a$ is nonempty (since it contains $\{\mathrm{t}\}$) and is closed under unions of chains. So by Zorn's lemma there is an $\mathscr{L}$-filter $F_a$ that is maximal with respect to the property of excluding $a$. $\bigcap_{a \neq \mathrm{t}} F_a = \{\mathrm{t}\}$. So $\boldsymbol{A}$ is a subdirect product of the quotient algebras $\boldsymbol{A}/F_a$, each of which is an $\mathscr{L}$-algebra. By the maximality of $F_a$, every $\mathscr{L}$-filter that properly includes $F_a$ must contain $a$. By the correspondence between the $\mathscr{L}$-filters of $\boldsymbol{A}/F_a$ and the $\mathscr{L}$-filters of $\boldsymbol{A}$ that include $F_a$ we get that the $\mathscr{L}$-filter generated by $[a](\Omega(F_a))$ is the smallest member of $\mathrm{Fi}_{\mathscr{L}}(\boldsymbol{A}/F_a)\backslash\{\{\mathrm{t}\}\}$. So each $\boldsymbol{A}/F_a$ is $\mathscr{L}$-subdirectly irreducible. □

Suppose $\mathscr{L}$ is defined by a set $\mathbf{L}$ of standard matrices. Consider any protomatrix $\boldsymbol{L}$ in $\mathbf{L}$ and let $\mathbf{K}_{\boldsymbol{L}}$ be the set of subdirect factors of $\boldsymbol{L}$ obtained from the proposition. Let $\mathbf{K} := \bigcup_{\boldsymbol{L} \in \mathbf{L}} \mathbf{K}_{\boldsymbol{L}}$. Since $\boldsymbol{L}$ is subdirect product of members of $\mathbf{K}_{\boldsymbol{L}}$, it is clear that $\models_{\mathbf{K}_{\boldsymbol{L}}}$ includes $\models_{\boldsymbol{L}}$ in the sense that $T \models_{\mathbf{K}_{\boldsymbol{L}}} r$ always implies $T \models_{\boldsymbol{L}} r$. Thus $\models_{\mathbf{K}}$ is included in $\models_{\mathbf{L}}$, which coincides with $\vdash_{\mathscr{L}}$. The inclusion in the opposite direction holds since each member of $\mathbf{K}$ is an $\mathscr{L}$-algebra. This gives the following:

**Proposition 5.8.** *Every standard autoalgebraizable logic $\mathscr{L}$ can be defined by a set $\mathbf{L}$ of standard $\mathscr{L}$-subdirectly irreducible matrices. Moreover, $\mathbf{L}$ can be taken to be the set of $\mathscr{L}$-subdirectly irreducible factors of the members of any given defining set of standard matrices of $\mathscr{L}$. Thus if $\mathscr{L}$ is defined by a finite set of finite matrices, then it can be defined by a finite set of finite, $\mathscr{L}$-subdirectly irreducible matrices.*

**Theorem 5.9.** *Let $\mathscr{L}$ be as in Theorem* 5.3. *Assume $\mathscr{L}$ is defined by a finite set $\mathbf{L}$ of finite, $\mathscr{L}$-subdirectly irreducible matrices. Then the special $\mathrm{ET}_{\mathscr{L}}$ algebras are exactly the $\mathscr{L}$-subdirectly irreducible $\mathrm{GET}_{\mathscr{L}}$ algebras.*

**Proof.** By definition $\boldsymbol{A}$ is an $\mathrm{ET}_{\mathscr{L}}$ algebra iff $\boldsymbol{A}_{\log}$ a protomatrix in $\mathbf{L}$; hence every $\mathrm{ET}_{\mathscr{L}}$ algebra is $\mathscr{L}$-subdirectly irreducible. Conversely, let $\boldsymbol{A}$ be an $\mathscr{L}$-subdirectly irreducible $\mathrm{GET}_{\mathscr{L}}$ algebra. By Theorem 5.3, $\boldsymbol{A}$ is isomorphic to a subdirect product of special $\mathrm{ET}_{\mathscr{L}}$ algebras and hence must be isomorphic to one of the factors of the subdirect product. □

We next show that most of the results concerning the specification of ET data types that were discussed at the end of Section 1 also apply, with certain modifications, to $\mathrm{ET}_{\mathscr{L}}$ data types over an autoalgebraizable logic defined by a set

of $\mathscr{L}$-subdirectly irreducible matrices. In the present context a *data structure* is a $\mathrm{GET}_{\mathscr{L}}$ algebra that is minimal in the sense that every element is denoted by at least one ground term. A *data type* is the isomorphism class of a data structure.

Recall that every universal initial specification of a classical ET data type is also a final specification and hence complete. An analogous result holds for $\mathrm{ET}_{\mathscr{L}}$ data types if we admit specifications from a somewhat wider class of formulas.

Let $E$ be a set of conditional equations and negations of conditional equations, all of signature $\Sigma(\Lambda)$. $E$ is an *initial specification* of an $\mathrm{GET}_{\mathscr{L}}$ data structure $\boldsymbol{A}$ if $\boldsymbol{A}$ is the initial algebra of the class of models of $E$. $E$ is a *final specification* of $\boldsymbol{A}$ if $\boldsymbol{A}$ is a final (terminal) algebra of the class of nontrivial data structures of this model class. $E$ is a *complete specification* of $\boldsymbol{A}$ if it is both initial and final.

**Theorem 5.10.** *Let $\mathscr{L}$ be as in Theorem* 5.3 *and assume in addition that $\mathscr{L}$ is defined by a finite set of finite, $\mathscr{L}$-subdirectly irreducible matrices. Let E be an initial specification of a special* $\mathrm{ET}_{\mathscr{L}}$ *data type $\boldsymbol{A}$ by conditional equations. Then there is a single ground term $t$ of signature $\Sigma$ such that $E \cup \{t \neq \mathrm{t}\} \cup \mathrm{AXGET}_{\mathscr{L}}$ is a complete specification of $\boldsymbol{A}$.*

**Proof.** By Theorem 5.9, $\boldsymbol{A}$ is $\mathscr{L}$-subdirectly irreducible. Let $a$ be a generator of the smallest nontrivial $\mathscr{L}$-filter of $\boldsymbol{A}_{\log}$. $a$ is denoted by some ground term $t$. Let $\mathbf{K}$ be the model class of $E \cup \{t \neq \mathrm{t}\} \cup \mathrm{AXGET}_{\mathscr{L}}$. $\mathbf{K}$ clearly contains $\boldsymbol{A}$, and in fact $\boldsymbol{A}$ is initial in $\mathbf{K}$ since it is initial in a larger class. $\boldsymbol{A}$ can have no proper homomorphic image in $\mathbf{K}$. For suppose it had one, say $\boldsymbol{B}$. Let $h: \boldsymbol{A} \to \boldsymbol{B}$ be the unique surjective homomorphism. ($h$ is unique because $\boldsymbol{A}$ is a data structure.) Let $F$ be the kernel of $h_{\log}: \boldsymbol{A}_{\log} \to \boldsymbol{B}_{\log}$. $F$ is an $\mathscr{L}$-filter since $\boldsymbol{A}$ is a $\mathrm{GET}_{\mathscr{L}}$ algebra. Thus $a \in F$ and hence $t^{\boldsymbol{B}} = \mathrm{t}$, contradicting the fact $\boldsymbol{B}$ is a model of $t \neq \mathrm{t}$. From the fact $\boldsymbol{A}$ is initial in $\mathbf{K}$ and has no proper homomorphic image in $\mathbf{K}$ it follows immediately that every data structure in $\mathbf{K}$ is isomorphic to $\boldsymbol{A}$. Hence $\boldsymbol{A}$ is final in $\mathbf{K}$. □

A set $\Gamma$ of universal sentences of signature $\Sigma$ is an *initial specification* of $\boldsymbol{A}$ if the set of conditional equations $\Gamma^* \cup \mathrm{AXGET}_{\mathscr{L}}$ is an initial specification of $\boldsymbol{A}$. (See Section 3 for the definition of $\Gamma^*$.) Under the hypothesis of the theorem, if a special $\mathrm{ET}_{\mathscr{L}}$ data type has a finite initial universal specification in this sense, then it has a finite complete specification.

**Theorem 5.11.** *Let $\mathscr{L}$ be as in Theorem* 5.10. *Every special* $\mathrm{ET}_{\mathscr{L}}$ *data type with a finite initial specification is computable.*

**Proof.** Let $\Gamma$ be a finite initial specification of an $\mathrm{ET}_{\mathscr{L}}$ data type $\boldsymbol{A}$. As in the proof of the last theorem, $\boldsymbol{A}$ has a complete specification of the form $K = \Gamma^* \cup \{t \neq \mathrm{t}\} \cup \mathrm{AXGET}_{\mathscr{L}}$. Because of completeness, the ground equations that hold in $\boldsymbol{A}$ are exactly

those ground equations that are logical consequences of $K$; hence they are recursively enumerable. On the other hand, the ground equations that fail in $\boldsymbol{A}$ are exactly those equations $r \approx u$ such that $t \approx \mathrm{t}$ is a logical consequence of $\Gamma^* \cup \mathrm{AXGET}_{\mathscr{L}} \cup \{r \approx u\}$; this is a consequence of premise $\boldsymbol{A}$ is $\mathscr{L}$-subdirectly irreducible and the choice of $t$. Thus the ground equations that fail in $\boldsymbol{A}$ are also recursively enumerable. It follows that the ground equations that hold in $\boldsymbol{A}$ are recursive, i.e., $\boldsymbol{A}$ is computable. □

It suffices for the conclusion of the theorem to assume only that $\boldsymbol{A}$ has a recursively enumerable initial specification.

## 6. Conclusion

The importance of nonclassical logics in computer science is a universally accepted fact, and algebraic data types with a nonclassical underlying logic seem to be worth investigating. The current interest in object-oriented programming gives added impetus to such a project. The use of McCarthy's 3-valued logic to formalize the way Boolean expressions are evaluated in some implementations can be viewed as the first step in forming a nonclassical data type. The next step would be to physically adjoin the 3-element algebra that defines McCarthy's logic to the data type as a new logic domain. Subsequent steps would involve defining the appropriate operations, such as the *equality-test* and *if-then-else* operations, that bind the logic domain with the other domains of the type. The act of actually adjoining the logic to the data type and adding the interconnecting operations is a big conceptual jump because it means moving the logic from the metalanguage to the object language, thus making it part of the programming language. It is problematical if there is merit in increasing the expressive power of the object language in this way. We do not confront this problem in this paper. To do this one would presumably have to investigate specific examples, like McCarthy's logic, in detail. But in order to give direction to these investigations, we feel it important to examine the role of classical logic in the theory of classical data types and to determine the scope of applicability of the classical methods. This has been the goal of the paper, and we now have a pretty good idea of both the power and limitations of these methods. The characteristic properties of classical logic in this regard are autoalgebraizability (the existence of an admissible term) and the deduction theorem. These are just the properties needed to obtain the classical-style conditional-equational axiomatization (Theorem 5.3) and the corresponding specification results (Theorems 5.10 and 5.11).

There are a large number of other questions we have not addressed here, at least not directly. Here are four that we think rank among the most important.

(1) When a logic, like McCarthy's, is embodied in a single, finite algebra it seems clear what form the data structure should take, at least in general terms. But what about a logic like $\mathscr{L}\mathscr{J}\mathscr{L}$ that cannot be defined this way? Should the basic data

structure in this case be the $\mathrm{GET}_{\mathscr{SSP}}$ algebras with $\mathbf{HE}_{\omega+1}$ as the logic domain, or rather their subdirect factors?

(2) The results obtained here can be viewed as an extension of closely related results for classical logic that are contained in [5, 15, 28, 33]. The main object of study in these latter papers are data structures with if-then-else operations that select a data object from two alternatives on the basis of an equality-test that must take one of the two classical truth values. Nonclassical if-then-else operations are not considered in the present paper. They certainly are important and deserve to be investigated.

(3) Limiting ourselves to standard logics seems too restrictive. For example we have already observed that the strong 4-valued Belnap logic considered in Section 2 is not standard. A smooth algebraic theory of nonstandard logics is available [4], but we do not know if the strong 4-valued Belnap logic falls within its scope. Possibly we should not even restrict ourselves to assertional logics. By taking the equational logic of an algebra for the underlying logic of a data structure, results analogous to Theorems 5.3, 5.10, and 5.11 can be obtained in a large number of cases, $\mathbf{KS}_3$ for example. However, if we take equational logic for the underlying logic of a data type, all queries, even those in the logic domain, must be in the form of equations. This does not seem very natural from a programming point of view.

(4) Finally, and most importantly, how should we handle logics such as both Kleene logics and the McCarthy and weak Belnap logics, that are not autoalgebraizable. Possibly this can be done by extending the signature by adjoining logical operations, like the equality-tests, as new operations. One has to be careful however that the underlying logic is not altered in the process.

## References


[1] R. Balbes and P. Dwinger, *Distributive Lattices* (University of Missouri Press, Columbia, MO, 1974).

[2] N.D. Belnap, How computers should think, in: *Proc. Oxford Internat. Symp. on Contemporary Aspects of Philosophy* (1975) 30-56.

[3] N.D. Belnap, A useful four-valued logic, in: G. Epstein and J.M. Dunn, eds., *Modern Uses of Multiple-Valued Logic* (D. Reidel, Dordrecht, 1977) 8-37.

[4] W.J. Blok and D. Pigozzi, Algebraizable logics, in: *Memoirs of the American Mathematical Society*, Number 396 (Amer. Math. Soc., Providence, RI, 1989).

[5] S.L. Bloom and R. Tindell, Varieties of "if-then-else", *SIAM J. Comput.* 12 (1983) 677-707.

[6] J.R.B. Cockett, Distributive logic, Technical Report, Computer Science Department, University of Tennessee, March, 1989.

[7] J. Czelakowski, *Model-Theoretic Methods in Methodology of Propositional Calculii* (The Institute of Philosophy and Sociology of the Polish Academy of Sciences, Warszawa, 1980).

[8] M. Dummett, A propositional calculus with denumerable matrix, *J. Symbolic Logic* 24 (1959) 97-107.

[9] H. Ehrig and B. Mahr, *Fundamentals of Algebraic Specification 1: Equations and Initial Semantics* (Springer, Berlin, 1985).

[10] M. Fitting, A Kripke-Kleene semantics for general logic programs, *Logic Programming* 2 (1985) 295-312.

[11] M.L. Ginsberg, Multi-valued logics, in: M.L. Ginsberg, ed., *Readings in Nonmonotonic Reasoning* (Morgan Kaufmann, Los Altos, CA, 1987) 251-255.

[12] J.A. Goguen, The logic of inexact concepts, *Synthese* **19** (1969) 325-373.
[13] J. Goguen, C. Kirchner, H. Kirchner, A. Mégrelis and J. Meseguer, An introduction to OBJ3, in: J.-O. Jouannaud and S. Kaplan, eds., *Proc. Conf. on Conditional Term Rewriting*, Orsay, France, 1987, Lecture Notes in Computer Science **308** (Springer, Berlin, 1988).
[14] J.A. Goguen, J.W. Thatcher and E. Wagner, An initial algebra approach to the specification, correctness, and implementation of abstract data types, in: R. Yeh, ed., *Current Trends in Programming Methodology* (Prentice Hall, New York, 1978) 80-149.
[15] I. Guessarian and J. Meseguer, On the axiomatization of "if-then-else", *SIAM J. Comput.* **16** (1987) 332-357.
[16] G. Grätzer, *Universal Algebra*, 2nd edn. (Springer, New York, 1979).
[17] D. Gries, *The Science of Programming* (Springer, New York, 1981).
[18] F. Guzman and C.C. Squire, The algebra of conditional logic, *Algebra Universalis* **27** (1990) 88-110.
[19] A. Horn, Logic with truth values in a linearly ordered Heyting algebra, *J. Symbolic Logic* **34** (1969) 395-408.
[20] S.C. Kleene, *Introduction to Metamathematics* (D. Van Nostrand, Princeton, NJ, 1952).
[21] K. Kunen, Negation in logic programming, *J. Logic Programming* **4** (1987) 289-308.
[22] J.-L. Lassez and M.J. Maher, Optimal fixedpoints of logic programs, *Theoret. Comput. Sci.* **39** (1985) 15-25.
[23] J. Łoś and R. Suszko, Remarks on sentential logics, *Indag. Math.* **20**, 177-183.
[24] J. Łukasiewicz and A. Tarski, Investigations into the sentential calculus, in: A. Tarski, *Logic, Semantics, and Metamathematics*, 2nd edn. (Hackett, Indianapolis, IN, 1983).
[25] B. Mahr and J.A. Makowsky, Characterizing specifications that admit initial semantics, in: *Proc. 8th CAAP*, Lecture Notes in Computer Science **159** (Springer, Berlin, 1983) 300-316.
[26] J.A. Makowsky, Why Horn formulas matter in computer science: Initial structures and generic examples, *J. Comput. System Sci.* **34** (1987) 266-292.
[27] J. McCarthy, Predicate calculus with "undefined" as a truth value, *Stanford Artificial Intelligence Project Memo* 1 (1963).
[28] A. Mekler and E. Nelson, Equational bases for if-then-else, *SIAM J. Comput.* **16** (1987) 465-485.
[29] E. Mendelson, *Introduction to Mathematical Logic*, 3rd edn. (Wadsworth & Brooks Cole, Monterey, CA, 1987).
[30] J. Meseguer, General logics, in: *Proc. Logic Coll. '87* (North-Holland, Amsterdam, 1989).
[31] J. Meseguer and J.A. Goguen, Initiality, induction, and computability, in: M. Nivat and J. Reynolds, eds., *Algebraic Methods in Semantics* (Cambridge University Press, 1985) 459-540.
[32] A. Mycroft, Logic programs and many-valued logics, in: *Proc. 1st STACS Conf.*, Lecture Notes in Computer Science 166 (Springer, Berlin, 1984).
[33] D. Pigozzi, Equality-test and if-then-else algebras: axiomatization and specification, *SIAM J. Comput.*, to appear.
[34] H. Rasiowa, *An Algebraic Approach to Non-Classical Logics*, Studies in Logic and the Foundation of Mathematics **78** (North-Holland, Amsterdam, 1974).
[35] N. Rescher, *Many-Valued Logic* (McGraw-Hill, New York, 1969).
[36] D.S. Scott, Some ordered sets in computer science, in: I. Rival, ed., *Ordered Sets* (D. Reidel, Dordrecht, 1982) 677-718.
[37] A. Tarski, Fundamental concepts of the methodology of the deductive sciences, in: A. Tarski, *Logic, Semantics, and Meta-Mathematics*, 2nd edn. (Hackett, Indianapolis, IN, 1983).
[38] J.W. Thatcher, E.G. Wagner and J.B. Wright, Specification of abstract data types using conditional axioms, IBM Research Report RC-6214, September, 1976.
[39] R. Wójcicki, Matrix approach in methodology of sentential calculi, *Studia Logica* **68** (1973) 269-279.
[40] R. Wójcicki, *Theory of Logical Calculi. An Introduction* (D. Reidel, Dordrecht, 1988).